CMP

ON THE FRONT LINE OF GAME INNOVATION

Game DEVELOPER

APRIL 2000

ARE YOU PREPARED FOR HARDWARE T&L?

PAGE 23 Manipulating Patch Surfaces

PAGE 42 Simulating a Ponytail, Part 2

PAGE 54 Postmortem: Turbine's ASHERON'S CALL

www.gdmag.com

U.S. $5.95 Canada $6.95

04>

0 73361 87387 3

DISPLAY UNTIL APRIL 24, 2000

Visit the brain behind the game at GDC 2000.

alienbrain®

Booth No. 1040, sponsored sessions are on March 10 and 11, 4-5 pm. Booth crawling

is also compulsary, so come and have a drink on us.

As a comprehensive solution alienbrain® provides a data management environment that tracks the content of your game, from creation to delivery. Version control, file sharing, annotations, searching, process automation and status tracking are just some of the features that allow your team to be as productive at the size of 20 as they were with 5, and work with 100,000 files as well as they did with 5,000.

genius | intelligence | control | knowledge

Game Developer front line AWARD

http://www.nxn-software.com
e-mail: info@nxn-software.com

NxN and alienbrain are registered trademarks

NXN

CONTENTS

VOLUME 7, NUMBER 4

APRIL 2000

FEATURES

32 All Aboard Hardware T&L

New graphics chips like the GeForce 256 accelerate transforms and lighting operations in hardware, allowing game scenes to be more visually complex and realistic. Fosner looks at hardware-accelerated T&L, explains what's involved from a game developer's perspective, and dives into the nuts and bolts of cubic environment mapping.

BY RON FOSNER

44 How to Simulate a Ponytail, Part 2

After leaving you hanging last month, Hecker winds up his series on open loop dynamic chains. Taking all of the tricky math and physics he had introduced, Hecker plugs and chugs, and out comes the basis for a realistically behaving ponytail.

BY CHRIS HECKER

54 Postmortem: ASHERON'S CALL

Creating a massively multiplayer game from the ground up with a team of uninitiated game designers sounds like a crazy idea. Teamed with Microsoft, the start-up crew at Turbine made modular architecture and dynamically load-balancing servers work like magic in ASHERON'S CALL.

BY TOBY RAGAINI

COLUMNS

15 Graphic Content BY JEFF LANDER
Lights...Camera...Let's Have Some Action Already!

23 Artist's View BY MEL GUYMON
Skin Deep 2: Implementing Patch Surfaces

80 Soapbox BY NOAH FALSTEIN
A Tale of Two GDCs

DEPARTMENTS

2 Game Plan BY ALEX DUNNE

4 Says You LETTERS FROM OUR READERS

7 Bit Blasts
Find Impulse's Illusion revealed, Sony gets it together for the PS2 launch, and Jeff Lander revisits Digimation's MultiRes Mesh.

COVER: *Depicting a valiant battle between a samurai knight and an orthlio from ASHERON'S CALL, this scene was created by the Turbine art team. For more information visit http://www.turbinegames.com.*

Game Developer (ISSN 1073-922X) is published monthly by Miller Freeman Inc., 600 Harrison St., San Francisco, CA 94107, (415) 905-2200. Please direct advertising and editorial inquiries to this address. Canadian GST: 124513185. **SUBSCRIPTION RATE** Subscription rate for the U.S. is $49.95 for twelve issues. Countries outside the U.S. must be prepaid in U.S. funds drawn on a U.S. bank or via credit card. Canada/Mexico: $69.95; all other countries: $99.95 (issues shipped via air delivery). Periodical postage paid at San Francisco, CA and additional mailing offices. **POSTMASTER:** Send address changes to *Game Developer*, P.O. Box 1274, Skokie, IL 60076-8274. **Customer Service:** For subscription orders and changes of address, call toll-free in the U.S. (800) 250-2429 or fax (847) 647-5972. All other countries call (1) (847) 647-5928 or fax (1) (847) 647-5972. Send payments to *Game Developer*, P.O. Box 1274, Skokie, IL 60076-8274. For back issues write to *Game Developer*, 1601 W. 23rd St. Ste. 200, Lawrence KS 66046-2703. Call toll-free in the U.S./Canada (800) 444-4881 or fax (785) 841-2624. All other countries call (1) (785) 841-1631 or fax (1) (785) 841-2624. Please remember to indicate *Game Developer* on any correspondence.

One Year Later....

It's difficult to believe that nearly one year ago the Columbine High School killings took place. In the wake of their violence, our industry (not to mention other media sectors) experienced close scrutiny from every corner in an attempt to establish a cause-effect relationship with this disaster. Today, we're still no closer to understanding the cause of that random violence, and I suspect we'll *never* know what motivated gunmen Harris and Klebold. However, despite all of the unanswered questions, I've been heartened by the way our industry has continued to evolve and shoulder the responsibility that is ours to carry. In the past year we neither compromised our artistic game design ideals, nor did we sit back uninterested. Instead, the industry continued to defend the rights of developers to express themselves while improving the ways that it alerts the public (parents in particular) about the content of games.

For me, the most affirming event of the past year was watching Doug Lowenstein, Chairman of the Interactive Digital Software Association, share the stage with Sen. Joseph Lieberman (D-Conn.) on C-SPAN and detail the strides that the industry made to improve the effectiveness of the Entertainment Software Ratings Board's ratings system, including expanded efforts to educate retailers and parents around the United States about the system. Lieberman, a longtime critic of violent videogames, was as upbeat about our industry as I've ever seen him.

Another point of progress over the past year was the ESRB's establishment of a new arm called the Advertising Review Council (ARC), charged to "ensure that advertisements placed by U.S. computer and videogame software makers are appropriate, responsible, truthful, and accurate." The ARC was announced in October, but the seeds of its creation really go back to 1998. Commenting on the ARC's establishment, Lieberman stated, "I want to applaud the videogame industry for taking seriously the concerns that were raised in the wake of Littleton and trying to do something about them. These measures, taken together, are a real gesture of responsibility, and an important step forward in our ongoing efforts to help parents protect their children from the harms of digital violence."

On one hand I'm glad to see Lieberman acknowledge our efforts, but I also believe that in the absence of Columbine and other senseless tragedies, we would have taken the same course of action. At last year's E3 I talked to many developers who felt the same sadness as the rest of country. These developers, like the vast majority of people in our industry, genuinely care about the effect that games have on children. Because of this, I'm confident that our progress in the areas of ratings and advertising are the result of our commitment to responsible actions, not knee-jerk reactions to violent events or political pressure. It's about us acting responsibly, and it makes me bullish about our future.

ADIEU, OMID. It's with regret that I bid columnist Omid Rahmat goodbye from the magazine. Omid has defected from our ranks to join Expertcity.com, where he'll be heading up business development efforts. We'll miss Omid and wish him all the best at his new .com job, and most of all we hope he remembers us when he's flush with IPO cash.

HELLO, CMP. You may have noticed a new logo on the upper left-hand corner of this month's cover. What's this "CMP" you ask? Last year the parent company of this magazine, Miller Freeman, acquired the large high-tech publishing company CMP Media, and beginning this month *Game Developer* will be published under the CMP brand. The CMP moniker will also attach itself to the other products and services we provide to the game development community, including the Game Developers Conference, Gamasutra.com, the GAMEXecutive Conference, the GDC HardCore Technical Seminars, and the Independent Games Festival, so you may seeing it quite a bit from now on. ■

Game Developer
ON THE FRONT LINE OF GAME INNOVATION
600 Harrison Street, San Francisco, CA 94107
t: 415.905.2200 f: 415.905.2228 w: www.gdmag.com

Publisher
Jennifer Pahlka *jen@mfgame.com*

EDITORIAL

Editorial Director
Alex Dunne *adunne@sirius.com*
Managing Editor
Kimberley Van Hooser *kvanhoos@sirius.com*
Departments Editor
Jennifer Olsen *jolsen@sirius.com*
News & Reviews Editor
Daniel Huebner *dan@mfgame.com*
Art Director
Laura Pool *lpool@mfi.com*
Editor-At-Large
Chris Hecker *checker@d6.com*
Contributing Editors
Jeff Lander *jeffl@darwin3d.com*
Mel Guymon *mel@infinexus.com*
Advisory Board
Hal Barwood LucasArts
Noah Falstein The Inspiracy
Brian Hook Verant Interactive
Susan Lee-Merrow Lucas Learning
Mark Miller Group Process Consulting
Paul Steed id Software
Dan Teven Teven Consulting
Rob Wyatt Microsoft

ADVERTISING SALES

National Sales Manager
Jennifer Orvik e: *jorvik@mfi.com* t: 415.905.2156
Account Executive, Silicon Valley
Mike Colligan e: *mike@mfgame.com* t: 415.356.3486
Account Executive, Northern California
Dan Nopar e: *nopar@mfgame.com* t: 415.356.3406
Account Executive, Western U.S. and Asia
Darrielle Sadle e: *dsadle@mfi.com* t: 415.905.2182
Account Executive, Eastern U.S. and Europe
Afton Thatcher e: *athatcher@mfi.com* t: 415.905.2323
Sales Associate/Recruitment
Morgan Browning e: *mbrowning@mfi.com* t: 415.905.2788

ADVERTISING PRODUCTION

Senior Vice President/Production Andrew A. Mickus
Advertising Production Coordinator Kevin Chanel
Reprints Stella Valdez t: 916.983.6971

MARKETING

Marketing Manager Susan McDonald
Marketing Coordinator Scott Lyon

CIRCULATION

BPA
Game Developer magazine is BPA-approved

Vice President/Circulation Jerry M. Okabe
Assistant Circulation Director Kathy Henry
Circulation Manager Stephanie Blake
Circulation Assistant Kausha Jackson-Crain
Newsstand Analyst Pam Santoro

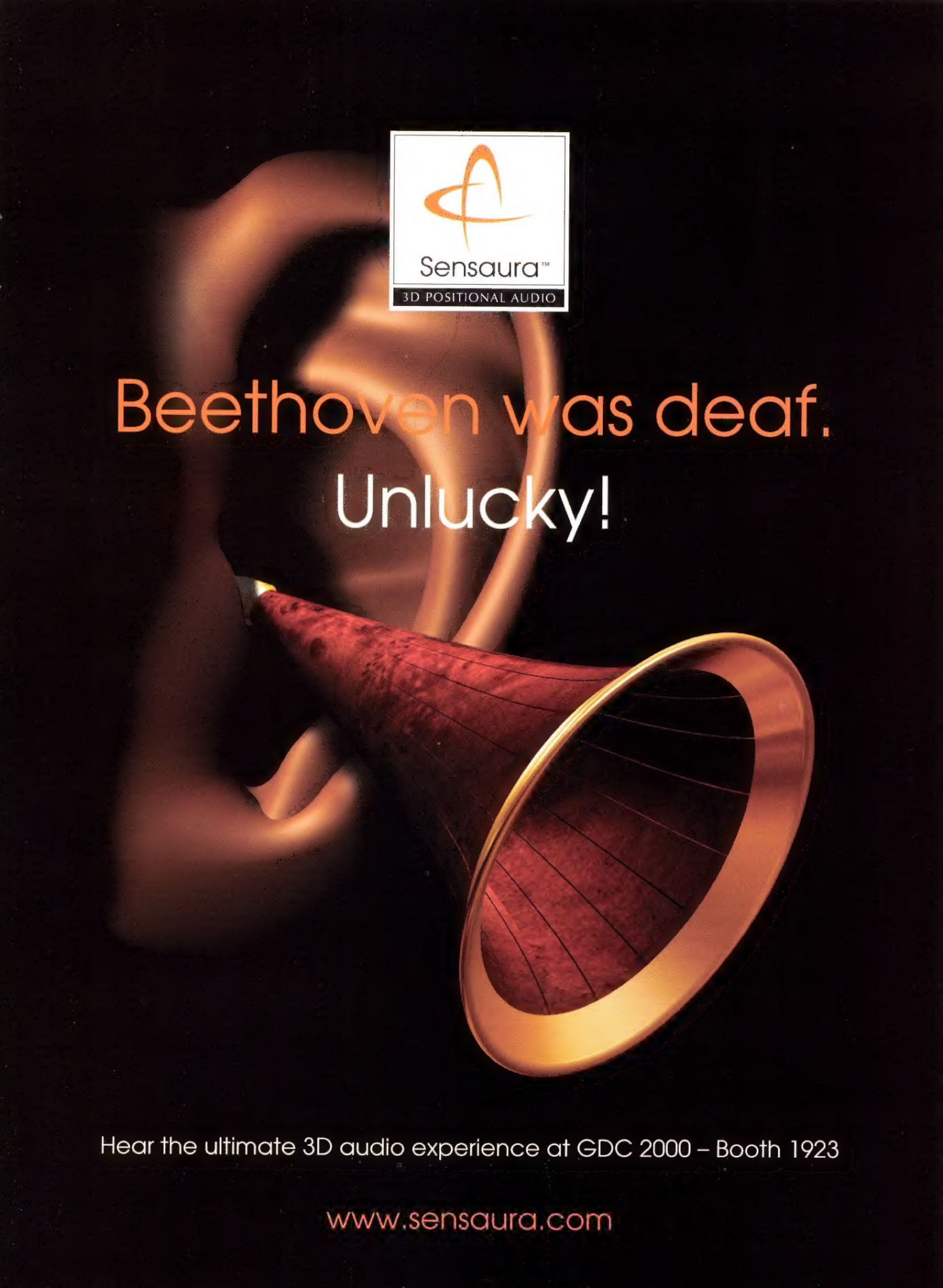
INTERNATIONAL LICENSING INFORMATION

Robert J. Abramson and Associates Inc.
t: 914.723.4700 f: 914.723.4722
e: *abramson@prodigy.com*

CORPORATE

President & CEO Gary Marshall
Corp. President/Business Tech & Channel John Russell
President/Business Technology Group Adam Marder
President/Specialized Technology Group Regina Ridley
President/Channel Group Pam Watkins
President/Electronics Group Steve Weitzner
General Counsel Sandra L. Grayson
Vice President/Creative Technologies Johanna Kleppe
General Manager/Game Media Division Greg Kerwin

CMP


Sensaura™
3D POSITIONAL AUDIO
Beethoven was deaf.
Unlucky!
Hear the ultimate 3D audio experience at GDC 2000 – Booth 1923
www.sensaura.com

Stop the Game, I Want to Get Off

I always enjoy reading the Postmortems in *Game Developer* to see the mistakes developers make, and also to get a window into the work practices of other games companies. In the Postmortem of AGE OF KINGS (January 2000), Matt Pritchard told us that developing the prequel AGE OF EMPIRES involved "working nonstop" for six months, which "took a heavy toll on people," so that the company decided to try to avoid this for AGE OF KINGS.

The way they tried to improve things was by spreading out the crunch times, and during crunch working 10 A.M. to midnight, Monday through Friday, with Wednesday nights ending at 7 P.M.

Is it just me or does this still sound pretty bad? I guess doing this for a week, or maybe two, wouldn't be so bad, but to do it for any longer periods, and repeatedly, sounds painful. Working these hours will strain your marriage, your family, and your friendships outside work. To attract and keep more experienced or specialist staff (who are often older, married, and have families), this would not be a recommended working practice.

As you work longer hours over prolonged periods you become weary and the quality of your code deteriorates. When games were smaller projects with one or two programmers, the quality of your code wasn't so important, you could just cobble it together. On larger projects as games are now, sloppy code will leave you increasingly drowning in bugs as you struggle to make headway at the end of your project which is "almost finished." These long hours will stretch your team relationships and tensions may develop, leading to unmotivated (and therefore less productive) staff, and people might start to leave. As an industry, can we please leave these barbaric and counterproductive methods behind?

Dr. Paul Tapper
Infogrames
via e-mail

Don't Gamble on Our Artistic Vision

I must come down firmly against content in *Game Developer* like Steve Boelhouwer's "Playing for Keeps: Developing Casino Games" (January 2000). The obvious rationale for including an article on gambling devices in the magazine is that they're games themselves, running on computers. This is where I think we need to be careful. The two industries (arcade gaming and slot machines) make different choices as to how they operate because they're fundamentally different.

We'll lend you an ear. E-mail us at editors@gdmag.com. Or write to *Game Developer*, 600 Harrison Street, San Francisco, CA 94107.

Any visual similarity doesn't change that. Transitively, this means that computer games and gambling devices are separate entities in separate industries.

While I agree that nobody can decide whether a game is or isn't "art," I believe that you can look at certain environments for game creation and say how much room for artistic innovation they offer.

Therefore, I feel pretty comfortable saying that developing the digital content for slot machines doesn't offer a very artistically fertile environment. (Note that I don't mean this in the graphical sense; graphically the slot machines are quite beautiful, I mean this in the sense of the game itself.) You have fairly ironclad bounds on how the game plays and works, and it corresponds to a fairly mundane system with a rather limited control set. Plus, you're catering to an audience for whom the maximum requirement is that they be able to put quarters in a slot; requiring these people to be capable of ten-move combos, fast reflexes, deep puzzle-solving, or army micromanagement is a losing proposition.

So you could ask, "So what? Why can't the magazine talk about opportunities that maybe aren't very innovative and don't offer much room for expression as game designers? It still offers opportunity, what's the problem?"

Here, I'll admit that I'm making some value judgments. But those value judgements boil down to this: I want *Game Developer* to be a magazine about innovation and advancement in game development. In times where there's such a push to drive our industry to crass commercialization, I think it's important to keep focused on the artistic qualities of our industry. You know, the qualities that keep us here instead of in other computer industries where the jobs are a little more secure and the pay's better. I'm not saying that money is evil, or that we should all wear berets and smoke cigarettes out of those stylus things. I'm just saying that unless we keep focused on the aspects of this industry that transcend money as an attractive force, our industry will eventually become a generic field of computing (just one that happens not to pay as well as others).

Brian Sharp
via e-mail

AUTHOR STEVE BOELHOUWER RESPONDS:

My purpose in writing this article was to shed light on an industry segment which many game development professionals may be only casually familiar with. You are correct that the design parameters and objectives for electronic gambling devices can be quite different from those of other game genres, just as an RPG has drastically different objectives from those of a first-person shooter. As a longtime reader of Game Developer, *I enjoy coverage of all of these genres, and would be disappointed if the publication were to narrow its focus and exclude certain aspects of our industry. You touched on career opportunities; this being a trade magazine (as opposed to a publication for game fans), I for one would like to continue to read about areas where our skills may be of value.*

With respect to the degree of innovation and "artistic fertility" possible in the industry, I would again opine that this is simply a function of the design parameters and objectives. The design and technology in modern gambling devices are more complex than perhaps you are aware. Furthermore, I disagree that "these people" (your description of casino patrons) are somehow incapable of enjoying advanced game-play features. In fact, a key design goal for the future will be to incorporate innovative features while still maintaining the fun and excitement of the gambling experience. We would welcome talented professionals such as yourself to join us in this task.

Game Developers Conference
March 10-12, 2000
San Jose Convention Center
California
Stand 940

Havok.com is the game technology wing of TeleKinesys Research Limited.

havok.com

USA
Phone **+1 408 376 4585**
Fax **+1 408 376 4586**

Europe
Phone **+353 1 677 8705**
Fax **+353 1 661 5200**

www.havok.com

WARNING

TAKE ONE PER GAME FOR PHENOMENAL PHYSICAL EFFECTS

VISUAL AND AUDITORY HALLUCINATIONS - COLD SWEATS - MEMORY LOSS - PALPITATIONS - CONFUSION - MENTAL DETACHMENT
EUPHORIA - NUMBNESS OF THE EXTREMITIES - PURPOSELESS MOVEMENT - SENSE OF WELL-BEING - INCREASED LEVELS OF SEXUAL DESIRE

While Others Tune Their Engines, Our Customers Cross the Finish Line

Only one 3D game engine has what it takes to put you on the fast track from Day 1: NetImmerse from NDL. Here's why developers worldwide power their games with NetImmerse:

SPEED & CREATIVE FREEDOM

"With NetImmerse, we can concentrate on the design of the game itself, reducing production time." – *Mark Jacobs, Mythic Entertainment*

"NetImmerse allowed us to focus our energies on the content elements that set Panzer General 3D Assault apart from the competition." – *Dan Cermak, Mindscape*

CROSS-PLATFORM

Supports PC, PlayStation® 2, MacOS, and Linux.

VERSATILITY

Delivers superior 3D graphics for any style of game (action, RPG, Internet and others) from any perspective.

SOURCE CODE AVAILABILITY

"NetImmerse's source code gives us the flexibility to adapt our applications in a number of different ways." – *Ron Fortier, Digital Domain*

TRACK RECORD

More games are in development or have been completed using NetImmerse than with any other independent 3D game engine – by far.

SUPERIOR TECHNOLOGY

"We are taking advantage of a broad range of features, including skinning for natural-looking characters, and continuous level of detail, dynamic multiple-color lighting, and multitexturing to create rich environments." – *Jean-Noel Portugal, Dramaera*

EXPERT DOCUMENTATION, SUPPORT

"...the best documentation I have ever seen for a game-related SDK, by a wide margin." – *Jonathan Blow, Game Developer Magazine*

"Support is always prompt. Usually the person who wrote the software in question is available to provide answers." – *Bob Case, AniVision*

Contact NDL today to find out about the fastest path from 3D game conception to completion.

NDL

www.ndl.com
650-328-4388
sales@ndl.com

Game Developer front line AWARD 1999

NETIMMERSE

Develop your game play, not your game engine.

Simon the Sorcerer 3D courtesy of Headfirst Productions
Prince of Persia 3D courtesy of Red Orb Entertainment
Panzer General 3D Assault courtesy of Mindscape SSI Division
Wild Wild West The Steel Assassin courtesy of SouthPeak Interactive
Wild Wild West™ ©1999 Warner Bros.
Racing car courtesy of AniVision, developer of *Net Race Live*

BIT Blasts

News from the World of Game Development

New Products: Impulse introduces Illusion for particle effects, Nichimen upgrades Mirai, and Criterion offers a Maya plug-in for Renderware 3. **p. 7**

Industry Watch: Sony consolidates in preparation for the PS2 launch, 3dfx trims the fat, and Monolith spins off Lithtech Inc. **p. 8**

Product Update: Jeff Lander reports on what Digimation and Intel have been up to since he reviewed the Multi-Res Mesh in November 1999. **p. 10**

New Products

by Daniel Huebner

Pixel Prestidigitation

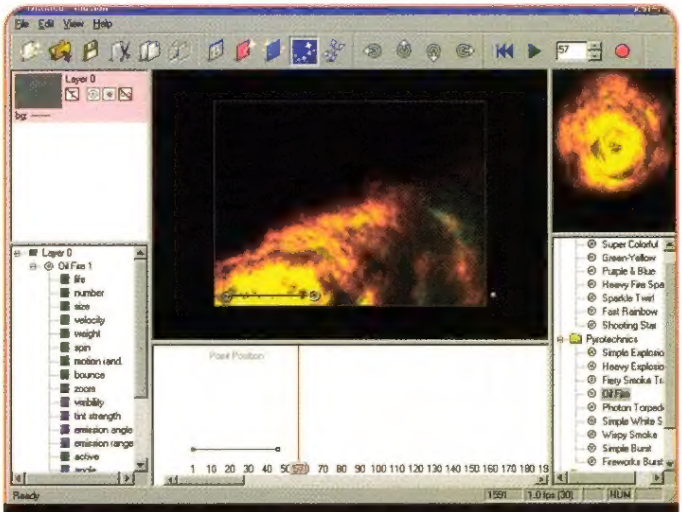
IMPULSE'S new Illusion: The Magic of Pixels is a particle effects system and compositing tool that aims to combine maximum control with absolute ease of use. Illusion allows users to work in a 2D environment with a WYSIWYG display showing what is going on during all work stages.

The key to Illusion's speed is its use of images to simulate a larger number of particles, reducing calculation and rendering time. Users can even use an .AVI or a series of images to create animated particles.

Effects can be added by selecting a specific effect from Illusion's emitter libraries and placing it on the stage. Many emitter properties can be manipulated directly from the main workspace, and low-level properties can be accessed through an emitter properties dialog while still providing a real-time preview of the changes. Illusion supports multiple layers and other functions to integrate effects easily into a 3D environment.

Illusion: The Magic of Pixels is priced at $249 and runs on Windows 95/98/NT 4. An OpenGL accelerator is recommended.

■ Impulse Inc.
Las Vegas, Nev.
(702) 948-1100
http://www.coolfun.com



Illusion's self-styled "WYSIWYG" interface offers real-time previewing of particle effects in a 2D environment.

Nichimen's High Hopes for the Future

MIRAI is Japanese for "prosperous future," and Nichimen Graphics has upgraded its modeling and animation software package of that name to ensure it meets its lofty goal. Mirai 1.1 is the next evolution of Nichimen's suite of real-time content creation tools, which replaced N-World as Nichimen's flagship package upon its original May 1999 release.

Mirai 1.1 contains enhancements to the render speed, display, and I/O of the original Mirai. The upgrade offers complete support for Game Exchange 2.1 as well as improved export to .OBJ and .3DS file formats. Geometry enhancements include additional default modeling options, improved camera manipulation, and magnet move multiple vertices on normals with falloff. Skeletal animation advances support magnet operations, squash and stretch, root rotation, and new display options and deformer icons. Mirai 1.1 includes support for additional domains including Sony, N64, and Dreamcast, and supports attribute translation between domains.

Mirai 1.1 is available for IRIX 6.3 or higher and Windows NT 4. It is priced at $6,495.

■ Nichimen Graphics Inc.
Los Angeles, Calif.
(310) 577-0500
http://www.nichimen.com

Breaking Down Development Barriers

CRITERION SOFTWARE has announced a new plug-in for Renderware 3 that can export world, object, and animation data created in Maya. Available for both PC and Playstation 2 platforms, the plug-in allows developers to build game levels and characters using a completely integrated package, modeling and animating in Maya while using Renderware 3 as the run-time engine.

The Maya exporter plug-in allows users to export entire 3D game worlds, complete with UV texture coordinates and vertex prelighting. The plug-in can also export textured objects and animations. Supported animation types include morph target, keyframe, animation sequence, and Renderware 3's proprietary "skin and bones" format. Criterion plans future support for Maya's procedural materials, skinning and weighting tools, and motion capture data in an upcoming version of the exporter plug-in.

The Maya exporter plug-in for Renderware 3 is included as part of the standard Renderware 3 SDK at no additional charge, and complete source code is provided to allow users to customize the tool.

■ Criterion Software Ltd.
Guildford, Surrey, U.K.
+44 (1483) 406233
http://www.renderware.com

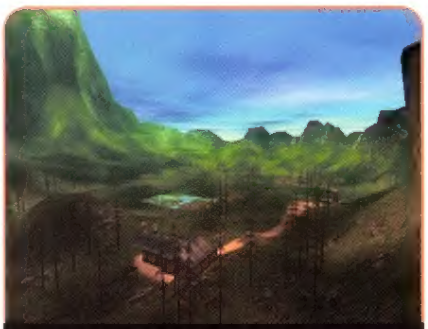
Industry Watch

by Daniel Huebner

SONY CONSOLIDATES. In preparation for the upcoming Playstation 2 launch, Sony Computer Entertainment of America is making moves to consolidate its operations. SCEA will merge spin-off developer and publisher 989 Studios into Sony Computer Entertainment America. Kelly Flock is due to step down as 989's president April 1, and leadership of the studio will fall to SCEA's management team with Kazuo Hirai as president and CEO of the entire North American operation. In addition, Shuhei Yoshida will join the merged organization as vice president of product development.

3DFX STREAMLINES. As part of an aggressive program to return to profitability, 3dfx Interactive announced approval of a plan to spin off its Specialized Technologies Group (STG) and reduce the company's overall workforce by 20 percent. The STG, which focuses on commercial multi-channel video and display, accounts for some of the workforce reduction, while other reductions will come from layoffs and attrition in redundant areas. Most affected will be 3dfx's administration, operations, sales, and software support departments. CEO Alex Leupp said the measures will help put the company in a more favorable position for its new fiscal year, which began January 1. One notable departure is the retirement of Bill Ogle from his position as executive vice president and vice chairman of 3dfx's board of directors. Ogle joined 3dfx as part of the company's 1999 merger with STB Systems, which Ogle co-founded in 1981.

MATTEL RESIGNATION. Mattel CEO Jill Barad resigned after failing to stem the decline of the company's sales and profits. Barad announced her resignation after Mattel announced its fourth-quarter losses in February. The company posted a net loss of $18.4 million on sales of $1.77 billion. Mattel had a net profit of $86.7 million on sales of $1.82 billion in the same period last year. Barad took the top job at Mattel in 1997 and oversaw Mattel's struggle with last year's $3.5 billion acquisition of The Learning Company. Mattel board members William Rollnick and Ronald Loeb took on the respective roles of acting chairman and acting chief executive as the company undertook its search for a new CEO.



The Lithtech engine is getting its very own company spun off from Monolith.

MONOLITH FOUNDS LITHTECH INC. Monolith Productions has announced a technology spin-off, Lithtech Inc., dedicated to the creation of real-time 3D development and networked multimedia operating systems. The company will become the caretaker of Monolith's Lithtech 3D engine technology licensing program, and has named 19-year Microsoft veteran Gregory Whitten as chief software architect. "Lithtech Inc.'s ability to attract someone of Dr. Whitten's caliber confirms our belief that our technology implementation is state-of-the-art," said Monolith Productions CEO Jason Hall.

SORENSEN LEAVES LUCAS. Jack Sorensen has stepped down as president of LucasArts Entertainment. At the time Sorensen made the announcement in February he did not specify what his future plans would be, saying only that he would take some time to work through his opportunities. The LucasArts board in turn promoted Simon Jeffery from his role as product development director to replace Sorensen as president. Jeffery joined LucasArts in 1998 to help expand the company's international business. "Former president Sorensen shaped the company's strategic direction, and I credit him with building LucasArts into one of the world's leading developers and publishers of interactive entertainment software," said LucasArts board chairman George Lucas. Sorensen's resignation follows closely on the heels of other high-profile departures from the company, including that of designer Tim Schafer. The LucasArts board also appointed May Bihir to the newly created position of vice president of worldwide sales and named Lucasfilm executive vice president Micheline Chau as lead director of the LucasArts board.

DAVIES DEPARTS DIGITAL ANVIL. Digital Anvil president Marten Davies left the firm February 1. Digital Anvil had originally envisioned itself as an independent publisher, but consolidation in the industry led the company to pursue development, leaving Davies with a less obvious place in the company's plans. Davies commented upon his departure that he had "enjoyed the challenge of assisting the company in laying the foundation stones for its current and future success," but he was "unable to bring any more immediate value to the equation." Digital Anvil's founder and chairman Chris Roberts said that the company would not name a successor; rather, it is dividing Davies's responsibilities among other members of the executive team. ■

UPCOMING EVENTS CALENDAR

3D Conference & Expo

SANTA CLARA CONVENTION CENTER
Santa Clara, Calif.
May 7–11, 2000
Expo: free w/early registration
Conference: $395–$895
http://www.3dshow.com

Electronic Entertainment Expo

LOS ANGELES CONVENTION CENTER
Los Angeles, Calif.
May 11–13, 2000
Expo: free to qualified industry professionals
Conference: $275–$450
http://www.e3expo.com

PlayStation is a registered trademark of Sony Computer Entertainment Inc.

PRODG

Hey thrill seeker, you've waited long enough. At last Pro-DG is here. Pro-DG is our sensational new development toolkit for PlayStation 2. It's completely customisable. It works at your speed, adapts to your methods and expands your capabilities. Pro-DG is really going to move the game on for you, so register for your free evaluation copy now at **www.snsys.com/gameon**

MOVING THE GAME ON

Digimation's Real-Time 3D Libraries

by Jeff Lander

In November 1999 in this very space, I took a look at the MultiRes Software Toolkit from Digimation. This product was created at the Intel Architecture Labs. And no, the gang up at Intel hasn't spent the last few months hunkered down in their Y2K bunkers. They have been cranking away on some great 3D technology.

This toolkit expands on the automatic level-of-detail (LOD) technology of the original toolkit by adding modules for subdivision surfaces, non-photorealistic cartoon rendering, skeletal character animation, and particle systems. These modules have been integrated into a single 3D rendering system. A sample application allows you to try out the various settings in each rendering module. These settings can then be used in your game project to achieve the same effect. Each module is sold as a separate license so developers can pick and choose only the pieces they really need for production.

MULTIRES 2. MultiRes 2 implements a continuous level-of-detail system for 3D objects. This system allows game programmers to dynamically scale the number of polygons in a model up or down to achieve the ideal speed vs. quality ratio for any rendering scenario. For example, when a character is farther away from the camera, the number of polygons can be reduced significantly, thus lowering the time needed to render that character. With the ability to adjust the polygon count of an object in continuous steps, the MultiRes system avoids the annoying "popping" effect seen when a model with a small number of discrete levels of detail is used.

The MultiRes Mesh component in the toolkit is largely unchanged from when I last looked at the package. The API has been changed slightly to integrate with the other pieces of the toolkit.

SUBDIV RT. The first new feature in the toolkit is a subdivision surface system. If you followed Brian Sharp's two-part series on subdivision surfaces (January and February 2000), you are already aware that this technology allows you to dynamically increase the complexity in a simplified basic mesh. This works by adding triangles to fill out the detail in a base mesh. For example, it will smooth out a curve by progressively adding more triangles to the object.

You can see this in Figure 1. In the simple low-resolution base mesh on the left, the detail is progressively added in the center mesh, achieving a halfway-point mesh and then finally a very detailed final model, shown on the right.

Using the SubDiv RT toolkit, you have a great deal of control over the subdivision method. One option is to decide to subdivide the mesh uniformly, so an equal number of triangles is created for each triangle in the original mesh. Another option is to use adaptive subdivision where the mesh divides to a level based on a user-supplied metric.

An interesting application for subdivision surfaces is using this technology in combination with the MultiRes system for online applications. In an online game situation, when a unique character is encountered, it may be necessary to download geometry for that character. By using subdivision surface technology, a very simple base mesh could be downloaded immediately to the user. This could be subdivided adaptively to make a nice-looking mesh. While this mesh is used, a higher-resolution MultiRes mesh can be downloaded in the background. Once it is available, the system can switch to the MultiRes system and use all that artist-created detail directly.

BONES RT. This package offers a real-time skeletal deformation system that can be used with both of the above rendering technologies. Using this system, a base mesh is deformed by a user-provided skeleton. The Bones RT toolkit provides algorithms to refine the weight system that attaches the skin to the skeleton. The user animates this skeleton and the Bones RT system deforms the skin automatically. By using quaternions to represent the orientation of each bone in the system, the animation frames can be interpolated smoothly to create nice in-between frames. You can see a sample creature with an embedded skeleton along with the skeleton control dialog in Figure 2.

One interesting addition to the skeletal system is the use of what is called BoneLinks. These are mini-bones that are

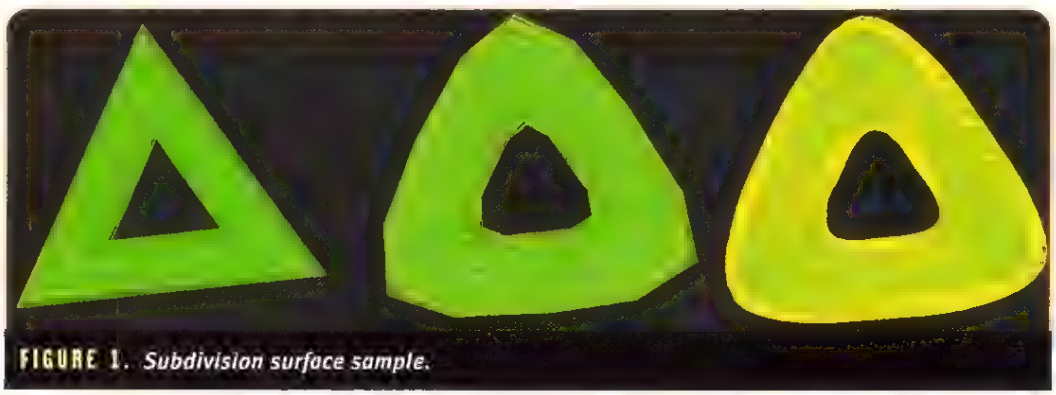


FIGURE 1. *Subdivision surface sample.*

Jeff has been writing way too much lately to be able to come up with a clever bio. Show him up by sending him something funny about yourself to jeffl@darwin3d.com.

MOTION CAPTURE ANIMATION TAKES AN EVOLUTIONARY STEP FORWARD.

Sophisticated motion capture for every production budget has arrived. MOTEK introduces UNICA™, the web-based animation solution that lets you rapidly custom-build and blend data sets from the world's largest motion capture library, view a preview, then purchase and download – at one-tenth the cost of original production. With UNICA, the world of motion capture is at everyone's fingertips. Come play on our site right now and watch your imagination take shape right in front of your eyes.

WWW.E-MOTEK.COM

603-641-1300
INFO@E-MOTEK.COM

Motek™
Technology in Motion

© 2000 MOTEK Motion Technology Inc. MOTEK, the MOTEK logo and UNICA are trademarks of MOTEK Motion Technology Inc

inserted between each bone at the joint to combat the problems of pinching and twisting that sometimes occur at skeletal joints.

RENDER RT. The Render RT module allows you to use non-photorealistic rendering techniques with a mesh generated by any of the other object tools. This module incorporates all of the ideas I discussed in my March column on cartoon rendering ("Shades of Disney: Opaquing a 3D World," Graphic Content, March 2000). Render RT detects silhouette edges as well as the edges defining material borders. These lines can be drawn with different colors, different thicknesses, and with or without antialiasing. The characters can then be shaded using different paint styles. The classic cartoon style allows you to define a shadow, highlight level, and threshold. This shading is applied to the base material color to create the "toon" look.

Beyond the cartoon shading, however, Digimation has added a sketch style that applies line textures to the object so it looks as if the character has been sketched. This is a multi-pass method that requires more rendering time, but it really looks different from traditional computer graphics, as you can see in Figure 3.

PARTICLE RT. The last module in the 3D Toolkit handles particle-system rendering. Unlike the other modules, this system doesn't really work directly with the other technologies but is more of a stand-alone particle system API. The API allows the user to control the generation, behavior, and look of the particles in the system. Forces can be created that interact with the system, such as wind and gravity. The particles can be constrained to follow a path and you can also create collision boundaries for the particles. By applying textures to the particles and controlling the scale dynamically, a great variety of effects can be achieved.

TOOLKIT SDK. The key feature of this toolkit is its API. The API allows developers to use this technology in their own projects. As with the MultiRes Mesh Toolkit, all of the run-time source for each module is included, so you can integrate the package easily into your own productions.

Each module also includes a sample viewer with source code along with documentation for both the API and the viewer.

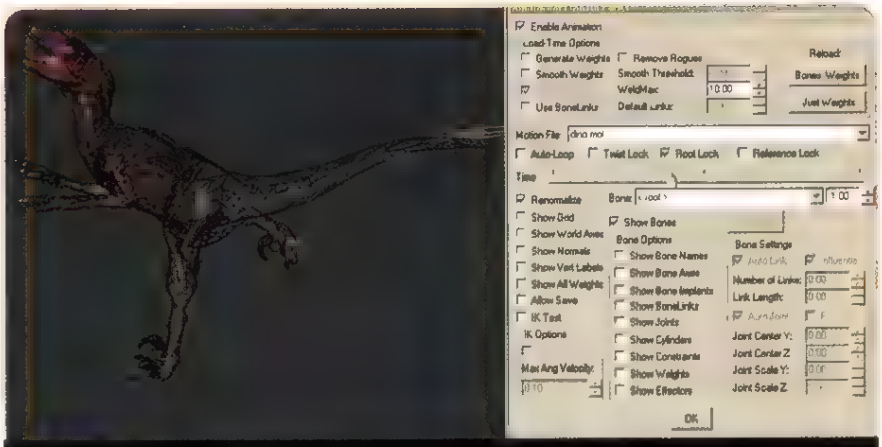


FIGURE 2. *Skeletal system and controls in the Bones RT module.*

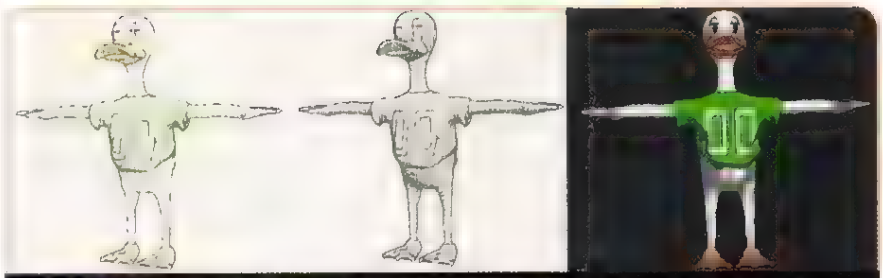


FIGURE 3. *The Render RT module adds a sketch effect to cartoon shading for a different look.*

PLUG-IN FOR 3D STUDIO MAX. To generate content for these systems, there is an export plug-in for 3D Studio Max. The plug-in allows you to define the API export parameters through a control interface. For users who do not work with Max, the API creation routines can be used with your own custom tools.

THE BOTTOM LINE. The Digimation Real-time 3D Libraries are a sophisticated collection of technology, and the source code is ready to be incorporated into a variety of 3D projects. It marks a significant enhancement of the original MultiRes 3D Toolkit. Each module is licensed separately so you are not required to buy something you are not going to use. If you are developing a 3D game project that requires cutting-edge 3D technology and looking for a head start, you should certainly check it out for yourself. ■

Digimation's Real-Time 3D Libraries with Intel Scalable 3D Graphics Software Technology:

Digimation Inc.
St. Rose, La.
(800) 854-4496

Software Requirements:
3D Studio Max for the plug-in;
Windows 95/98/2000/NT 4.0;
Run-time code is portable to other platforms.

Pricing: Each module price is given per finished game title.

MULTIRES 2 RT: $10,000 including three copies of the Max plug-in.

SUBDIV RT: $10,000

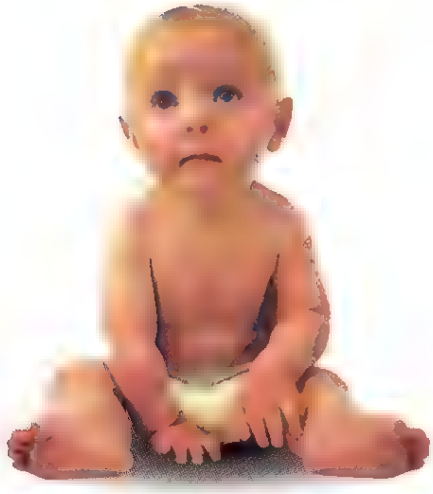
BONES RT: $10,000

RENDER RT: $5,000

PARTICLE RT: $5,000

MULTIRES 2 PLUG-IN FOR 3DS MAX: was $295, currently priced at $99

If only someone could give me a tool that would let me design fast network games with drastically reduced development time and cost so I'd be free to express my creativity to its fullest potential and improve my time-to-market, then I'd be happy.



I'll smile now!

We've got what you've been dreaming of. Introducing NetZ, a powerful new toolkit for creating high-performance, multi-player network games. Sure to put a smile on your face. Because with NetZ minding the network, you get to spend your time on game design. NetZ gives you fault-tolerance, load balancing and dead-reckoning right out of the box. Internet game design just got a whole lot easier.

[Proksim]

Shaping the online future

Phone: (514) 940-9090 info@proksim.com www.proksim.com

Come see us at GDC 2000 in booth 1536

MotionStar

Don't just make it

move, Make it Groove!

Make your motion Smooth with MotionStar Real-Time Motion Capture.

Now with popular plug-ins, it's easier than ever to capture high-quality motion in every major animation program! Ascension gives you:

- Complete turn-key systems to take motion capture to final production.
- Real-time, smooth choreography of movements.
- Wireless technology to really *groove*, baby, *groove!*

Add MotionStar to your studio's toolbox: Your creativity and Ascension's MotionStar are essential for today's animator! **Call today to see how an in-house motion capture system from Ascension can work for you. Ask for our free demo reel!**

MotionStar®
REAL-TIME MOTION CAPTURE

Ascension
Technology Corporation

Call. **800-321-6596** *Outside N. America.* **802-893-6657**
Visit our web site at. **www.ascension-tech.com**
e-mail: ascension@ascension tech.com *Fax* 802 893-6659
PO Box 527, Burlington, VT 05402 USA

Visit us at Booth #737 at GDC

©1999 Ascension Technology Corporation
MotionStar is a trademark of Ascension Technology Corporation. Megadude ©1998 Protozoa, Inc.

Lights...Camera...Let's Have Some Action Already!

EXT. STREET — NIGHT — WIDE HIGH ANGLE

Camera tracks BURKE as he walks down rain-drenched street. Light foot traffic. BURKE nods to a doorman and continues, turning right on Catalina Street.

CUT TO:

EMPTY SIDE STREET — MED. FULL

Camera tracks BURKE walking slowly up Catalina. Footsteps can be clearly heard following. BURKE notices and eyes dart but does not turn. Shot widens as he continues up street. ROCCO enters frame following BURKE casually strolling along smoking. BURKE turns to a stop in front of a restaurant, "examining" the menu. ROCCO comes to a stop in a doorway, puts out his cigarette with his shoe.

CUT TO:

REVERSE — OTS TWO SHOT — MED. FULL

BURKE turns continuing down street. ROCCO starts to follow again. BURKE grimaces as footsteps resume. Camera tracks ahead of BURKE showing ROCCO over shoulder. Continues leading him until he comes to a newsstand. He stops and turns to pick up and examine a paper, footsteps slow behind him coming to a stop.

CUT TO:

MED. CLOSE — BURKE

He picks up a magazine, eyes alert. Trying to catch a glimpse of his pursuer to his right out of the corner of his eye. Suddenly a hand grabs his left shoulder. He jumps.

CUT TO:

TWO SHOT — BURKE AND STAND OWNER

STAND OWNER
I'm not running a library here, Mac. That'll cost you a nickel.

BURKE takes a beat, tucks the paper under his arm, and tosses the man a coin. Turns to continue down street. Footsteps continue behind him. He quickens his pace. The footsteps quicken. He breaks into a run. His pursuer starts to chase.

ULTRAVIOLENT TERRIBLY FASCINATING STORY BEGINS...

From Big Screen to Game Screen

All right, so it's a cliché action sequence. I'm a programmer and this is not my latest screenplay. But that's not the point. A quality director can take these simple ideas and create tension, drama, and anticipation. These are exactly the qualities that pull people into a story. However, in interactive game applications, generating feelings of tension and drama is very difficult.

It's easy enough to create a cinematic cutscene that follows traditional filmmaking techniques. However, this yanks the player out of the interactive experience. Modern 3D game engines can create cinematic sequences within games, but most of the time these sequences are completely scripted using traditional animation techniques. The sequence fires when the player enters a location, pulls a lever, or triggers some other mechanism. Once started, the sequence follows a deterministic path. The game designer now has a choice to make. The first option is to control the camera shots to show the drama, suspending the interactivity. Second, the player can maintain complete camera control and try to catch the action. This can create a great sense of "What's going on up there?" as you rush to find a viewpoint. HALF-LIFE used this technique very effectively. However, crucial information cannot be delivered in this manner as the player may miss it by spending too long studying the magnificent architecture.

The ideal solution would be to present the drama to the character as much as possible while allowing the player full control. It's clear to me that the camera system in a story-driven game needs to be a crucial character. It needs to be aware of what the player is doing, what is going on in the world. The camera needs to be "intelligent" enough to find the best viewpoint to show the player what's going on without ruining the dramatic element. To address this situation, I'm going to explore the idea of "camera AI."

Smart Cameras

Conventional game wisdom seems to hold that 3D real-time shooters must use a first-person camera while story-driven 3D action and adventure

Jeff is the lead programmer and chief idea monkey at Darwin 3D, but what he really wants to do is direct. When not out pitching his latest spec script, he can be harassed at jeffl@darwin3d.com.

games need to use a third-person camera. This may be the case. It is certainly true that the first-person point of view (POV) is not an effective movie storytelling paradigm. If you get the chance to see the film *Lady in the Lake* directed by and starring Robert Montgomery, certainly check it out. It is a very interesting moviemaking experiment. Montgomery filmed almost entirely from a subjective point of view. The only time you see the protagonist is in reflection. While compellingly different and not a bad movie in its own right, it shows dramatically why the first person is not the most effective method for conveying drama.

Take the rough scenario I outlined at the beginning of the column. Played in the first person, I would hear footsteps and need to swing the camera around to catch what was going on. All the subtlety would be gone. My shadow would either simply be hidden when I turned, or be caught diving behind a wall, cover blown. Likewise, for the newsstand sequence, I would need to rely completely on sound to convey the surprise from the stand owner grabbing my shoulder. While all these issues have solutions, the first-person POV certainly limits the possible options. With this in mind, let's take a look at automated methods for third-person POV cameras.

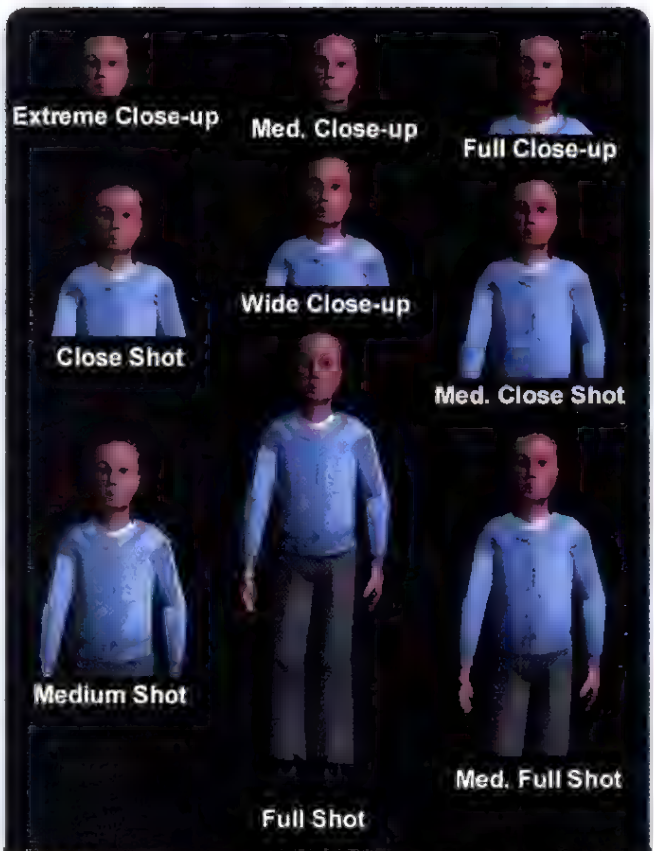


FIGURE 1. *An array of commonly used terms for describing shot selection.*

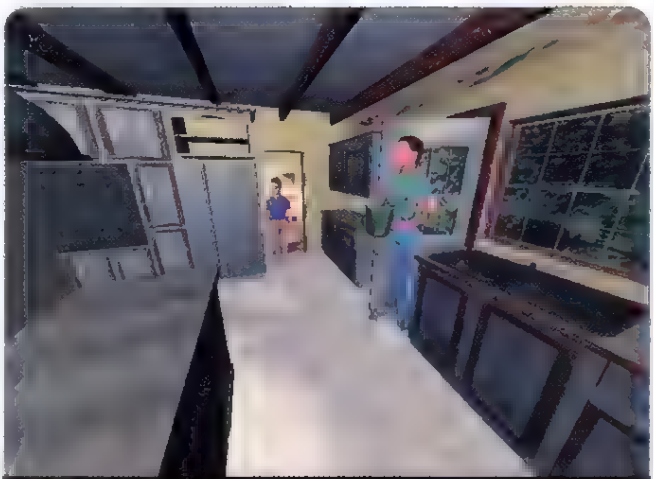


FIGURE 2. *The fish-eye effect can be an unintended result of incorporating too much width in the field of view.*

The Shooting Gallery

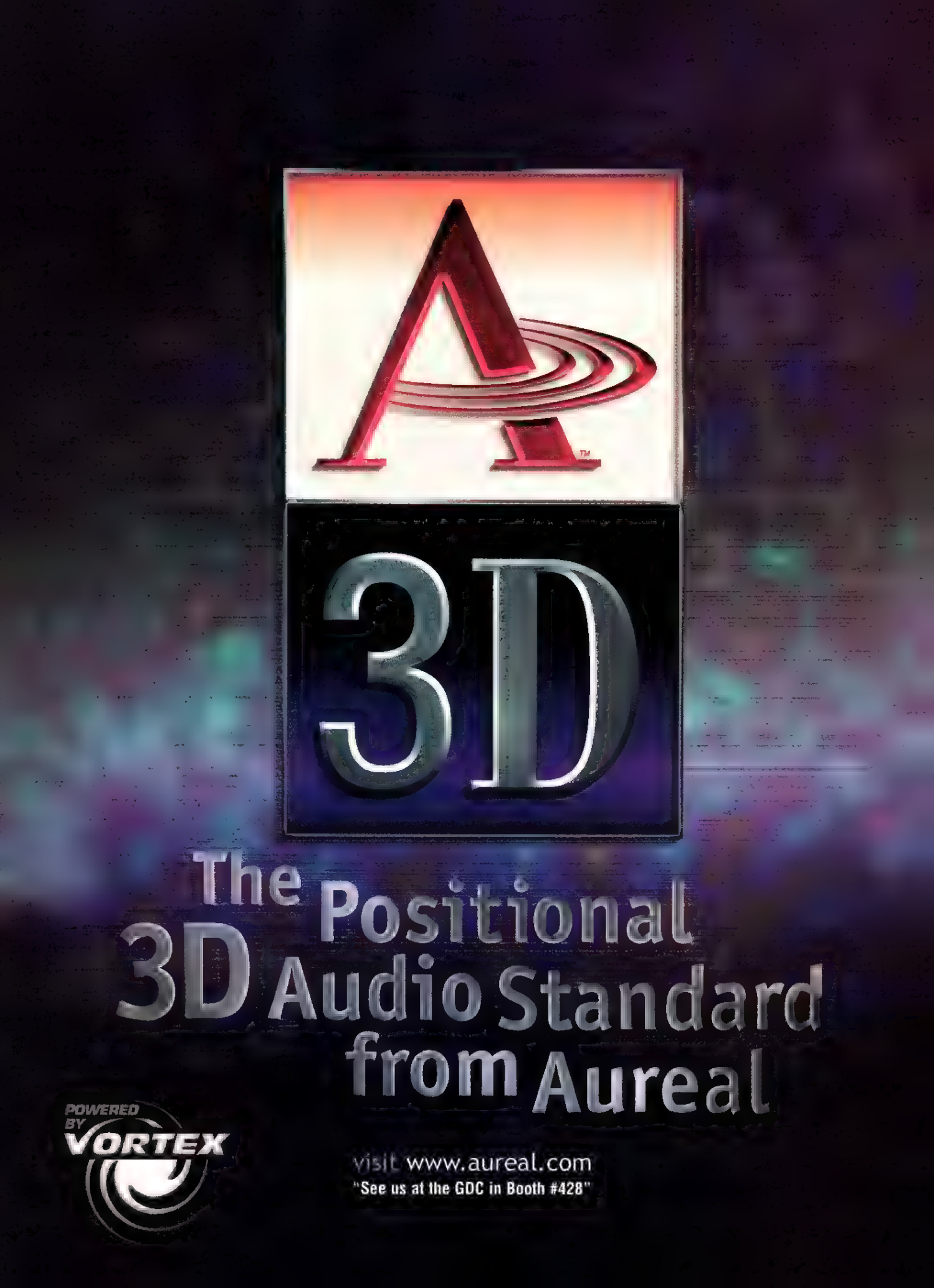
To begin with, I need to create a frame of reference for all the shots I want to compose. Fortunately, many years of cinematography have provided a ready-to-use guideline for shot composition. Let me start with the framing of a shot for a single person. Obviously, the simplest step would be to divide the shots into long, medium, and close. A long shot would include the character and the environment, a medium shot might be the character from the knees up, and a close-up would be just the head.

This doesn't really give the fine-grain descriptive terms that would be useful for framing a character. Typically, cinematographers frame the human figure using nine basic shots. The terms I am going to use are:

- Extreme close-up
- Medium close-up
- Full close-up
- Wide close-up
- Close shot
- Medium close shot
- Medium shot
- Medium full shot
- Full shot

You can see the framing for these shots in Figure 1. Once defined, these different shots are easy to work with in a real-time 3D game scenario. The goals are to center the character on the screen at the proper distance for the various shots. The first thing to find is the camera target, where I am going to point the camera. I could track a series of focus points on each character. Fortunately, I embed a skeletal system inside my characters so I can animate them. The base of each bone in my skeletal system is a ready-to-use focus point. I just pick the appropriate bone base to "look at" and use that as the camera target. For some of these shots, the focus point will be in between bones, but it's easy enough to interpolate the position between them.

Getting the right framing once I have the correct focus point is a bit trickier. I have two parameters I can play with. I can change the distance of the camera to the character or I can change the field of view (FOV) on the camera, effectively zooming in or out on the character. I have found that game players are very sensitive to FOV changes. If the field is too wide, the view takes on a fish-eye lens look (Figure 2) which can be very annoying (or cool, depending on your needs). If the field is too narrow, objects are hard to keep centered as subtle movements are exaggerated by the extreme zoom (think sniper rifle). So, I try to stay away from adjusting the FOV whenever possible. It's much better to move


3D
The Positional 3D Audio Standard from Aureal
POWERED BY VORTEX
visit www.aureal.com
"See us at the GDC in Booth #428"

the camera. However, because we are in an interactive world, the players can move around into situations where the camera is in a tight squeeze and pulling back is not possible without breaking through part of the set. Here, a little zoom now and then is a good thing.

So, using these simple guidelines, I started to create some "camera AI" routines. Given a character and a desired framing, the camera routine will start moving the camera to the desired position. This gives me the ability to say, "I'm ready for my close-up, Mr. De Mille," and it happens — all without any keyframes or scripting. But monologues are not very exciting.

Let's Strike Up a Dialogue

Adding a second character to the mix complicates things quickly. Let's consider a scenario where a character walks up to another and starts talking. I start by tracking the main character with a full shot, as in Figure 3 for example.

As my character walks along, the other character can either approach me, or I can approach and start talking. At this point a dialogue is initiated and I need to start considering both actors in the scene as a pair. There is an imaginary line connecting the two. This line is actually very important as it defines a vertical plane that a camera cut between two cameras should not cross in most situations. I have seen several games and even some movies make this mistake. "Crossing the line" can really disorient the viewer because when a camera cut crosses the line, the relationship between the parties changes. The character on the right is suddenly on the left and your mind doesn't immediately follow the motion path of the camera.

I start off with an establishing shot that shows the special relationship between the two characters. If I'm lucky, the standard single-player camera shows the character adequately and can be used as an establishing shot. If for some reason the view of the second character is blocked, the camera needs to swing around to frame both characters in the view. I just spin the camera around the key character until both are in view. At this point, I can set up my dialogue cameras.

For two characters in a dialogue, I set up a bunch of virtual cameras that I can cut between while the dialogue takes place. All of these cameras are on the same side of "the line." I decide which side to stay on based on the initial positions, the direction each character is facing, and the environmental restrictions. Usually, there is a preferred side that is obvious given the conditions. The cameras I set up, shown in Figure 4, are:

1. Group profile: camera perpendicular to the line, framing both subjects.
2–3. Individual profiles: one camera framing each character.
4–5. OTS: over-the-shoulder shots of each subject.
6–7. Reaction shot for each subject.

Now, that looks like a lot of cameras. However, one thing I always find amazing about 3D games is that cameras are so underutilized. From a technical perspective, cameras are dirt cheap. A position, orientation, and field of view are all you really need. There is absolutely no reason for games to use the same camera, panning, swiveling, and gliding everywhere. Camera cuts are a very important part of storytelling. When was the last time you watched a film or television show that used a steady-cam following everyone around all the time? (Well, don't count the opening sequence of *Touch of Evil*.) That is what we have currently in most games. (All right, mini-rant mode off.... Now back to our story.)

FIGURE 3. *Adding a second person requires careful shot selection to help viewers understand what is going on.*

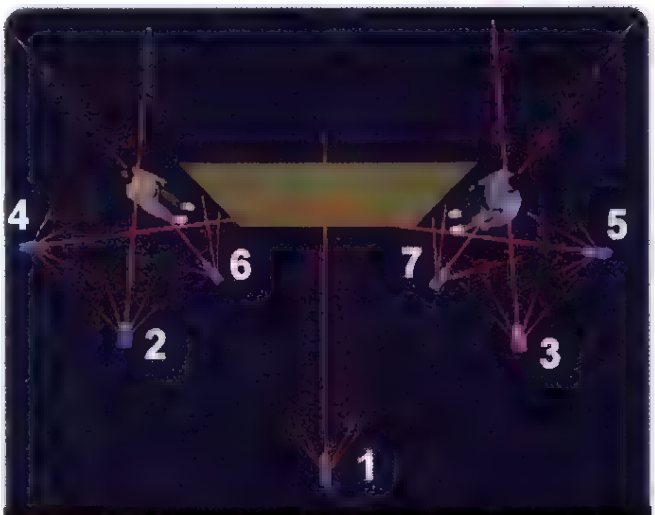


FIGURE 4. *A camera setup for a two-person dialogue, offering a variety of camera cuts for different moods and effects.*

Using the action line between the characters and the character positions, these camera positions are calculated using simple 3D math. As a guideline, I have found that about ten degrees off the action line is good for the OTS cameras and 60 degrees is good for the reaction shots. The other cameras are just perpendicular to the line. You remember how to take the perpendicular to a vector, right? (Big hint: Swap X and Y and negate one.) During the scene, the action line can move if the participants move. Anytime that happens, the cameras just get recomputed.

Once all the cameras are set up, I need to determine which ones to use. This is where the camera AI comes in. The camera system needs to know about the characters. Who is talking? What's the emotional state of each character? For example, when a

Atari designed him to kill without mercy.

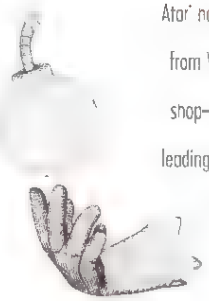
We taught him to throw a nice curveball.

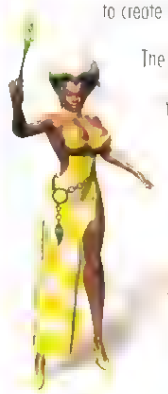
W W W . V I E W P O I N T . C O M / A T A R I

Meet the Green Jester—a fuse-happy fool with an explosive arsenal. When the other heroes from Atari's hot new *Gauntlet: Dark Legacy* need some comic relief, this is the clown they turn to. And when the game's production team needed to beat some crazy deadlines, they turned to Viewpoint.

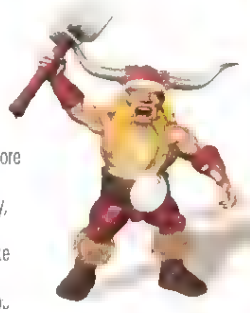
Having worked on previous *Gauntlet* characters, we were excited to create 3D models for *Dark Legacy's* lead warriors. The bomb-flexing Jester, however, seemed a little stiff. You see, his original design didn't include any finger action, which meant his trademark toss looked...well, funny.

"One of Viewpoint's designers spent his *day off* building this skeleton for the Jester's hand," says Atari's Scot Caterson, the game's art director. "We hadn't even specified it that way, he just thought it would look more realistic. When we saw it, we agreed."





Atari has come to expect this kind of enthusiasm from Viewpoint. After all, we're not just a model shop—we're a Strategic 3D Resource, used by more leading developers than just about anyone else. Hey, we taught this jolly green joker to throw like a pro. Just think what we could do for you. Call 888-368-6073 or visit our website, and learn how you can get better 3D, with less fooling around.



Viewpoint DIGITAL™

A Dassault Systèmes company

BEHIND THE SEEN

© 2000 Viewpoint Digital, Inc. All trademarks or trade names used herein belong to their respective companies.

character is talking, you probably want to use either the OTS camera or the profile camera that shows the speaker. While the character is speaking, you may occasionally want to switch to a reaction shot, particularly if the "mood" of the character dramatically changes based on the AI, script, or whatever. This is also where that real-time facial animation system you invested in earns its pay. Believable emotional reactions will really build the drama of the scene.

I used a very interesting reference source (see "The Virtual Cinematographer" in For Further Info) to set up a finite state machine that decides which camera to switch to depending on factors similar to the above, as well as delay timers. Another very good idea they suggested is to prompt the AI of the characters to move a little if they are too far apart or are blocking the camera. Though I haven't tried that yet, it certainly makes a lot of sense.

Two's Company, Three's a Crowd

Extending this idea to more than two subjects complicates the setup quite a bit. For three people, a line can be established between two of the participants based on who is speaking and who is facing one another. The general patterns for a three-person conversation will fall into an "A" or "L" pattern depending on the layout of the participants and the action line chosen. In Figure 5, the people are in an "A" pattern leaving a nice action line between speakers A and C. However, there are also valid action lines between the other players. So what do you do? You can always transition to the establishing shot to reset the group. In general, however, when the action line changes, there will be a valid camera that was on the same side as the previous line, so that should be the first choice. You can further complicate these groups by adding individual reaction shots. However, the basic alignment should provide enough options.

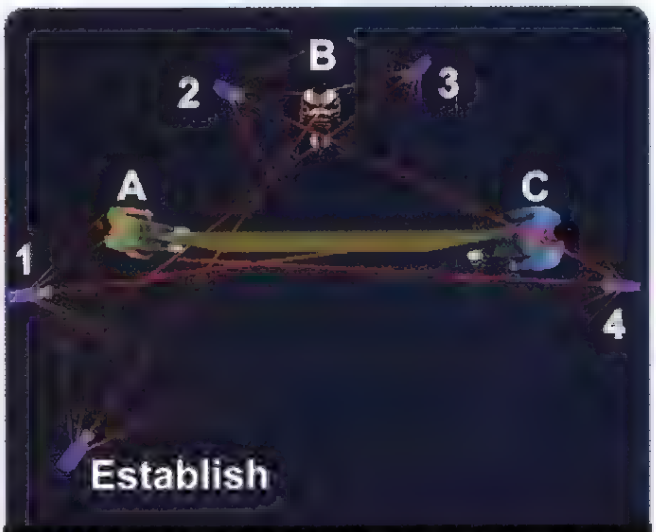


FIGURE 5. *Adding a third person complicates matters but shouldn't disrupt the action if cameras are set up properly.*

For more than three speakers, it's usually best to generalize the shots into group shots. You can create close-ups for the speaker and try to cut that with OTS shots from different sides of the group. Luckily for game developers, large groups of characters aren't things we want a whole lot of anyway, for other reasons.

Let the Player Have Creative Control

Everyone wants to be a director. This includes the game player. Most players will want to control the action. However, these techniques give the player a great deal of options beyond just spinning the camera around the key character, hoping to find a good view. You can allow the player to jump through your possible cameras, modify the view once the cut happens, or take total control to prevent cuts. They could even assume a first-person POV, if desired. The point is to provide options that allow you to tell a story without locking the player out.

Another interesting point is that the participants do not necessarily have to be people with whom the player speaks. When the player goes to pick up an object, open a door, or look at a painting, the object in view can become one of the subjects in the dialogue. Looking over the shoulder of a character at a painting and then cutting back for the reaction shot would be very cool. Think of the AI reactions you could fire off.

Our games are becoming more complex, and it is definitely time to start thinking about more sophisticated camera usage. Applying some of the AI techniques we have been using for characters to cameras certainly makes a lot of sense. As a bonus, experimenting with these sorts of camera routines is pretty fun. It's amazing how you can begin to influence mood and pacing by manipulating very few parameters. ■

For Further Info

- Bares, William, Joël Grégoire, and James Lester, "Realtime Constraint-Based Cinematography for Complex Interactive 3D Worlds." Proceedings of the Tenth National Conference on Innovative Applications of Artificial Intelligence, Madison Wis., July 1998. pp. 1101–1106. Available at http://www.[illegible]/lester/www_media_papers.html.
- He, L., M. Cohen, and D. Salesin. "The Virtual Cinematographer: A Paradigm for Automatic Real-Time Camera Control and Directing." *Proceedings of Siggraph*. New York: ACM Siggraph, 1996. pp. 217–224.
- Katz, Steven D. *Film Directing: Shot by Shot*. Studio City, Calif.: Michael Wiese Productions, 1991.

More Research on Cartoon Rendering

After my non-photorealistic rendering columns in February and March, I got a note from Adam Lake, one of the researchers at Intel who is working on NPR. He is presenting a paper at the Non-Photorealistic Animation and Rendering Symposium this summer covering more advanced silhouette and shading algorithms. He has graciously made it available to the public. You can get the paper at http://www.[illegible].

Acknowledgments

Thanks to Lisa Washburn at Vector Graphics ([illegible]) and Steve Tice at QuantumWorks Corp. ([illegible]) for art contributions.

SEE US AT GAME Developers Conference 2000 BOOTH # 538

BOPS

BILLIONS OF OPERATIONS PER SECOND

At the Core of DSP Innovation

BOPS ENABLES

- Quick Time-to-Market, highest performance, scalable and reuseable DSP IP cores
- Rapid development with world class compilers and software development tools
- High-Volume SOC solutions for Visualization, Networking and Communication markets
- Seamless integration with ARM™ and MIPS® processors in SOC's

TRANSFORM YOUR WORLD AT WWW.BOPS.COM

info@bops.com • 650.330.8400

it's all about control...

SynthCore™ SDK

Sonic Versatility for game sound developers

Physical Modeling - Event Modeling - DLS & Wavetable engine

www.staccatosys.com

Hear us at GDC Booth 840

Skin Deep 2: Implementing Patch Surfaces

Last month we examined NURBS, patch, and subdivision surfaces as methods for creating real-time 3D characters. This month, we'll pick up where we left off by putting the theory to practice and examining how to implement the patch surfacing method with RT3D geometry.

As I discussed in my previous column, there are many reasons to make the shift from polygonal modeling to one of the various surfacing techniques available to us. As the technology for processing and rendering 3D content continues to advance, so too does our ability to create realistic and engrossing 3D content. However, to take advantage of the increased capability, the content we generate must be correspondingly more complex. More complexity means more polygons, and more time spent building, texturing, and animating them. In order to avoid ever-increasing development times, we as developers need to identify those processes that can be made "complexity-independent." That is, we need to evolve our content creation methods so that we feel free to increase the detail and diversity of our virtual worlds, and do so within our allotted development cycle.

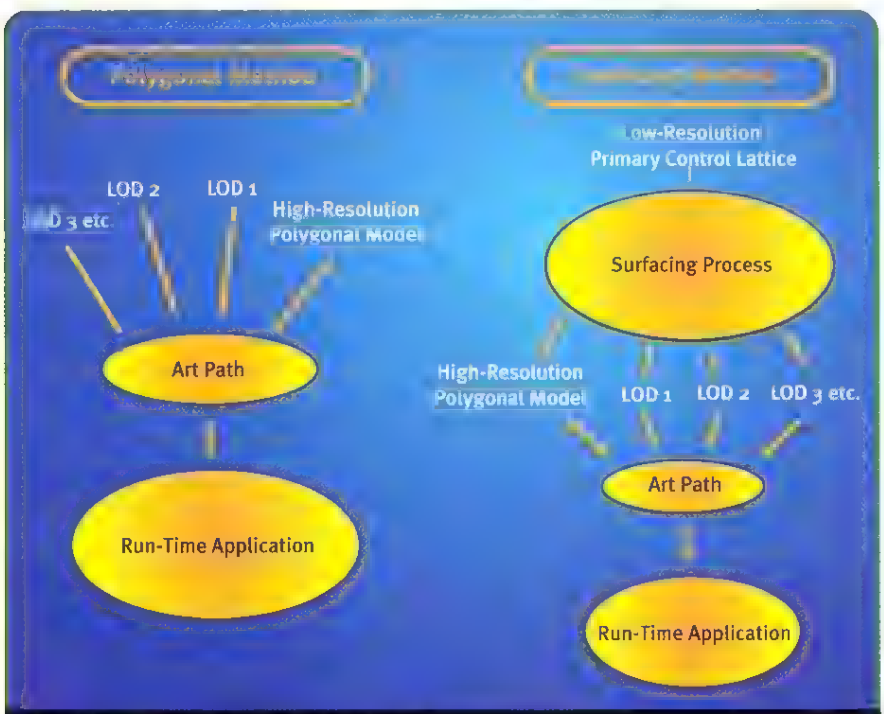


FIGURE 1. *Control point methodology.*

Mel Guymon has been animating in the gaming industry for several years. When he's not at his desk pushing polygons, he can usually be found at the local Barnes and Noble, slumming for reference materials. Mel can be reached at mel@infinexus.com.

In response to this, recall that last month we set out to define a process by which an artist could generate more complex models without spending too much time working with large amounts of data (Figure 1). In this ideal process, the artist, working only with a relatively low-resolution control point lattice and relying on a procedural method for extracting the high-resolution surface at the output, could generate arbitrarily complex models while maintaining a rapid and efficient workflow. To satisfy the requirements of this ideal process, we identified three potential surfacing methods: NURBS, patch, and subdivision surfaces. Though all three of these methods met the basic criteria as outlined in our ideal process, my choice turned out to be patch surface modeling (for details on all three methods, see "Skin Deep: Surfacing Strategies for RT3D Characters," Artist's View, March 2000). Right now, the best implementation of patch surface modeling can be found in 3D Studio Max 3.

Patch Modeling Basics

In Max, the patch surfacing technique combines the best things about polygon-mesh and NURBS-surface generation. As is the case with polygonal modeling, artists can control the process at the very lowest level of data, and can create their surfaces one patch at a time if necessary. Once a patch is created, the tools for manipulating it and its subobjects (edges and vertices) are directly analogous to polygons. Patches can be extruded and tessellated, beveled, detached, and so on.

FIGURE 2. *Example patch shapes.*

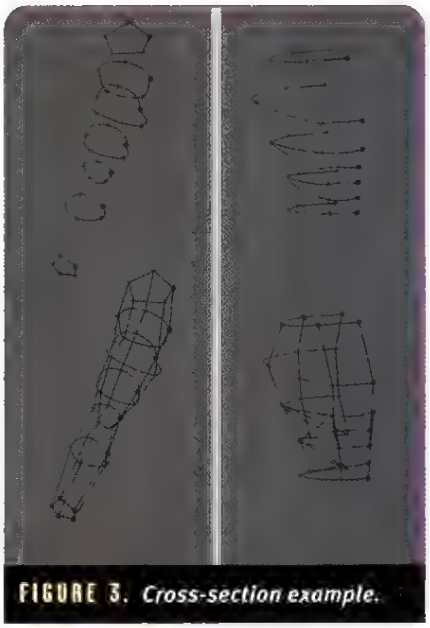


FIGURE 3. *Cross-section example.*

As is the case with NURBS surfaces, the surface detail information is stored in a lattice of B-splines, whose curvature can be adjusted readily by a set of control-point tangent handles. Moreover, the surface resolution can easily be scaled up and down without affecting the underlying topology. Thus, smoothly organic shapes can be created efficiently, without the artist having to depend on complicated NURBS techniques or complex subdivision surfaces.

All B-spline patch surfaces use as their building blocks patches generated with three- or four-point polygons or splines. Figure 2 shows some example splines and the resulting surfaces. Note that in the third example, the spline lattice has not been completely partitioned into three- or four-point polygons. As a result, there are gaps in the resultant patch surface. This is due to the fact that in Max there are only two types of patches, tri-patches and quad-patches. Note also that with each increase in the number of steps, the curvature of each shape becomes more refined, as delineated by the control-point tangent handles. By using these control handles, the artist can create an almost limitless number of variations in the topology of the surface, all without ever increasing the density of the governing control point lattice.

Surface Tools

The primary method for creating patch surfaces in Max is through the use of the Surface Tools. These include the Cross Section and Surface modifiers. The Cross Section modifier is used to connect a set of splines defining the cross-section of a polygonal object. This is necessary because a spline lattice composed only of cross-sections has, by default, not been partitioned in the requisite three- and four-point polygons necessary for surfacing. In Figure 3, you can see an example of how this works. On the top left, we see a set of splines defining the cross-section of a human arm. Below that is the same set of splines after the Cross Section modifier has been applied. On the top right you can see a second example where the number of vertices in each cross-section differs. Below that, the Cross Section modifier has again been applied, though this process is somewhat hit-or-miss, since the modifier has no way of intelligently determining where best to place the connecting splines.

The Surface modifier converts a spline lattice into a patch surface by creating a patch at every point where spline segments intersect to form a three- or four-point polygon. For further editing, the resulting patch surface can then be modified with an Edit Patch modifier, which gives the artist access to the entire set of patch editing tools.

Converting a Polygonal Mesh Directly to a Patch Surface

Although the preferred method for creating a patch surface is through the use of the Surface Tools, it's possible to convert an existing polygonal mesh to a patch surface directly. This is done by applying an Edit Patch modifier to a polygonal mesh, or by collapsing a polygonal mesh to an Editable Patch object. The main advantage of using this method is that any polygonal object can be converted into a patch surface. And since in many cases the data has come from an outside source (such as from a digitized model), the only current way to see the data is in polygonal form. Additionally, existing polygonal models that were created before the artist had access to the patch surface method can be converted immediately to patch surfaces. Finally, the vast majority of RT3D artists have experience working only with polygons, and as such there may be a slight learning curve associated with the Surface Tools process.

The main disadvantage to this process is evident in Figure 4, where

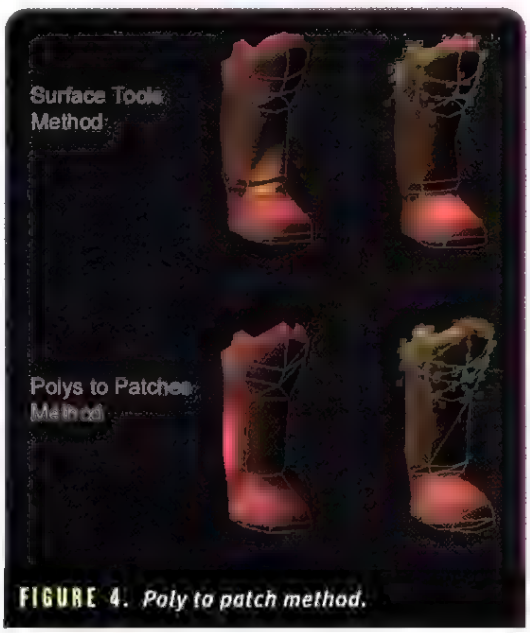


FIGURE 4. *Poly to patch method.*

We Make Motion Capture Perform

Above—Melvin: The Redheaded Stepchild

Download Free Motion Samples

Our proprietary MotionReality tracking and analysis software allows digital artists to capture and implement complex motions faster than ever before. Whether it's a little boy wailing on his favorite teddy bear or the lightning fast moves of martial arts warriors, Giant Studios can capture the action without losing the nuances of the motion. See for yourself, download the sample motion data on our website.

Soccer Player w/Ball

Martial Arts Warriors

See us at booth 847 GDC 2000. Look for the Ant

GIANT STUDIOS

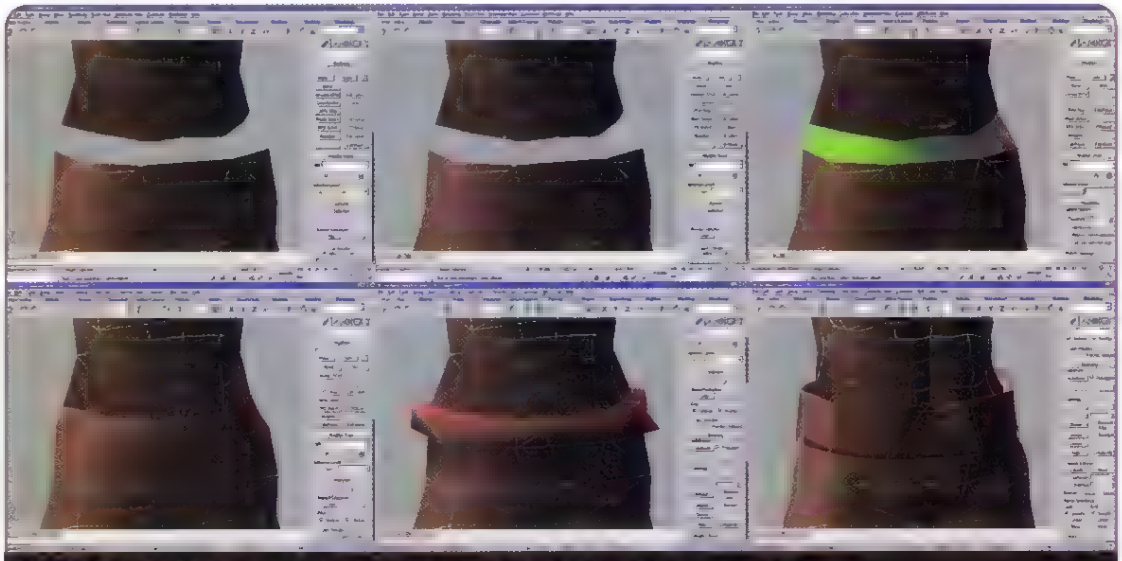


FIGURE 5. *Adding a belt solves the problem of joining two disparate surfaces.*

two surfaces with identical topology have been created. On the top we see a spline lattice and the resulting patch surface created with the Surface Tools method. On the bottom is a polygonal mesh and the resulting surface created by adding an Edit Patch modifier. If you look closely, you will notice that there are many more control handles on the patch surface on the bottom right. This is because at every control point, the number of tangent handles corresponds directly to the number of edges intersecting that point. When the polygonal mesh is converted to a patch surface, the result is composed entirely of tri-patches. This results in a maximum number of edges with a maximum number of tangent handles, and the absolute least efficient surface for the corresponding geometry.

Alternately, polygonal data generated from an outside source can be used as a template on which a spline lattice can be built. This technique can prove to be much faster than one might think, since by using the 3D Snap tool, the spline network can be laid down very quickly. And though there is a learning curve associated with the Surface Tools method, in fact the similarities to polygonal modeling are so great, and the resulting advantages so significant, there is little question that the effort of learning the patch surfacing process is time well spent for any RT3D modeler.

Example Flowpath: Stitching Two Surfaces Together

One problem that has always been hard to work around is how to connect two dissimilar organic surfaces, such as an arm to a shoulder or a torso to a pair of legs. With patch surfaces, however, the advantage comes in the fact that the source data is of extremely low resolution. In Figure 5, we see two surfaces that need to be joined together to create a single continuous surface. Note, however, that where the two surfaces are to be joined, they differ in the number of vertices. To accommodate this, we'll need to create an in-between surface. With the 3D Snap function turned on, a spline lattice is created with an upper and lower cross-section corresponding exactly to the surfaces to be welded (top left). Then the cross-sections are connected with individual spline segments (top center). Finally, a Surface modifier is applied, creating a patch surface which fits both the upper and lower surfaces (top right). These can now be welded together to create a single continuous surface. Alternately, we could have simply attached the two objects to each other and created the in-between patches one at a time by using the Add Tri or Add Quad functions of the Edit Patch modifier.

Continuing on in Figure 5, we see that the models have been welded together, and the curvatures adjusted through use of the control handles (lower left). To add a belt to this character, we need only to select a set of patches and extrude them, in much the same way as we would extrude a set of polygons. Following along in the figure, we see that the patches are first selected, then extruded, and finally expanded using the Outline function of the tool.

The similarities to polygonal modeling are so great, and the resulting advantages so significant, there is little question that the effort of learning the patch surfacing process is time well spent for any 3D modeler.

Noncontinuous Surfaces

One subtle limitation for patch surfaces is that when a patch is subdivided, all the patches welded to it subdivide as well. This can be a problem if the model you're working with has areas of both high and low detail. This is the case particularly with humanoid characters, which can have areas of intricate detail around the face and hands, and areas of comparatively lower detail, such as in the arms, legs, and torso. Figure 6 shows an example of this. In this model, the head is fairly detailed while the rest of body is sim-

Congratulations!

She thinks your new game is awesome.

Even better that she got it for free!

Surprised? You shouldn't be. With the popularity of new CD-R drives and the Internet, it happens all the time. And it costs you plenty. In fact, casual copying of CD games can reduce 7-10% of your revenues.

That's why you need SafeDisc. It's the best way to slow down piracy and prevent casual copying from eroding your profits

It's fast. Protecting your rights doesn't take long and it doesn't interfere with game playability.

It's easy. We have over 80 replicators licensed all over the world ready to work with you to protect your latest creation.

It works. SafeDisc-protected titles are being distributed by major publishers throughout North America, Europe and Asia — over 30 million disks last year alone

Take steps to protect your profits with SafeDisc and burst the bubble of casual copiers once and for all.

Call **1-800-900-4229**, or visit our website at **www.macrovision.com** for more information

MACROVISION

Protecting your image

1341 Orleans Drive
Sunnyvale, CA 94089
Phone: +1 (408) 743-8600
Fax: +1 (408) 743-8610
Web: www.macrovision.com

© 2000, Macrovision Corporation. SafeDisc is a trademark of Macrovision Corporation and C-Dilla, Limited

See us at the Game Developer Conference, Booth #1142

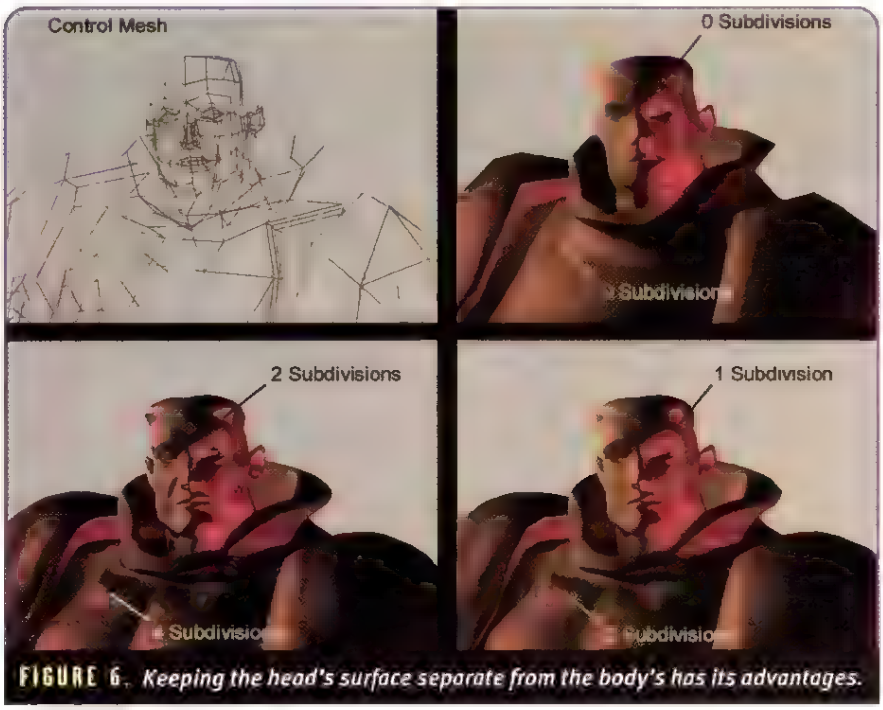


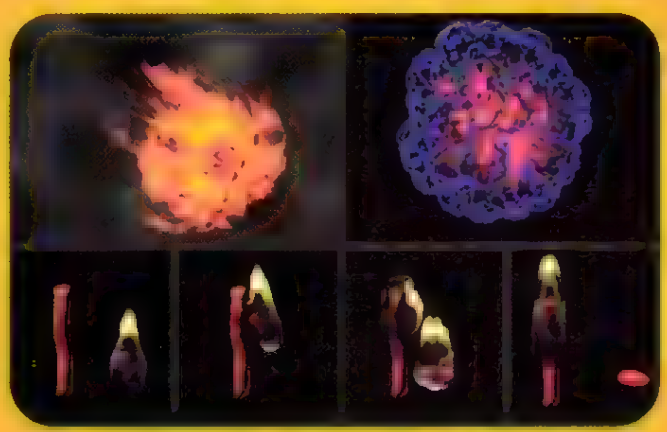
FIGURE 6. *Keeping the head's surface separate from the body's has its advantages.*

plistic by comparison. As a result, when the model is subdivided, the head will increase in complexity much faster than the rest of the model. In order to avoid this, the head has been kept as a separate surface, not welded to the torso. At the point where the neck joins the torso, there will be a discontinuity, but this has been hidden beneath the collar on the model. Furthermore, by keeping the surfaces separate, it is possible to implement object-swapping for equipment adjustment, level-of-detail (LOD) implementation, or selective morph-target animation. The control point lattice on the top left has been surfaced using the Surface Tools method. On the top right, the image is rendered with zero subdivisions, which is equivalent to a normal polygonal model. Continuing around the image in a clockwise direction, the model has been subdivided with an increasing number of subdivisions, with the head object containing one and two subdivisions, while the body goes through three and four levels of subdivisions respectively. Note the high amount of complexity within the head with only one or two subdivisions. The polygonal complexity of the head is roughly 220 quad polygons, with the next two levels at 1,100 and 2,400 quad polygons respectively. The high amount of detail is a characteristic of patch modeling and is by far the most impressive visual property of this

PLUG-IN POWER!

DIGIMATION'S PHOENIX EFFECTS

Back in February ("Pyro-Techniques: Playing with Fire," Artist's View), I discussed the advantages of prototyping in-game effects through the use of procedural effects such as Maya's Particle FX and 3D Studio Max's Combustion plug-in. One of the downfalls of procedural routines is that they fail to capture the subtle variations in shape, color, and density that are characteristic of most forms of combustion. For this reason, artists often rely on footage of real-life conflagrations upon which to base their effects. The downside of this is that unless you are generating the footage yourself, you pretty much have to settle for whatever you can get your hands on from a third party. With procedural effects, the artist has the advantage of being able to prototype and tweak the look of the effect directly. This has two clear benefits. The first, obviously, is that the artist is able to customize the effect in accordance with his or her needs without relying on an external source. The second, more subtle benefit is that by using a procedural routine to prototype the in-game effect, the artist gains experience working with a particle system interface. This experience will enable the artist to contribute to the design and implementation of the in-game particle system.



As particle system technology continues to advance, more and better particle routines are becoming available for use in game development. One of the most recent particle system plug-ins is Digimation's Phoenix for 3D Studio Max 3. Similar in execution to the Combustion plug-in that ships with Max, Phoenix generates a volumetric effect at render time, capitalizing on Max's native particle technology. As you can see in the image at left, Phoenix is capable of generating a wide variety of realistic effects, from chaotic fireballs and luminous volumetric plasma, to a single flame on a matchstick. The interface is straightforward and functional, and the plug-in ships with a wide variety of preset effects routines, covering most of those required in game development. The product is currently only available for 3D Studio Max, for around $400. (For more information go to .)

surfacing technique, highlighting the artist's ability to store massive amounts of information within the low-resolution mesh.

The Mad Scientist at Work

In Figure 7 we see another, more striking example of the scalability of an in-game character model generated using the Surface Tools technique. At the lowest resolution, the character comprises roughly 1,800 three-point polygons, with the next two levels of subdivision at 6,000 and 23,000 polygons respectively. Note again that the control point lattice for this character is extremely low-resolution when compared with the final high-resolution model, and that due to the patch surface technique, these variations in complexity can be achieved totally on-the-fly simply by ramping up or down the number of subdivisions in the patch surfaces.

FIGURE 7. *The Mad Scientist shows off his scalability.*

Get the Skinny

From a content-production standpoint, the advantages of patch-surface modeling are significant enough to warrant a change of methodology for most RT3D applications. However, an even larger benefit is seen once the content actually makes it into a 3D engine which has been optimized for rendering patches. The most basic advantage from a rendering standpoint is that the resulting surface has a regular, consistent structure, optimal for use in the ultrafast tri-strip rendering methods. Additionally, since there is a relatively small amount of data compared with the final topology, time spent on data transfer and geometry transformation is minimized. As a result, in most applications, switching from polygons to patches can double the resulting frame rate. Coupled with the resolution-independent nature of the content creation process, the advantages in rendering speed and data manipulation should enable the patch surfacing technique to maintain a solid footing in the RT3D world. That is, until the next big surfacing technology breakthrough comes along. ■

Acknowledgements

Special thanks to Beau Perschall and Jeff Yates. The Mad Scientist character was designed and modeled by Frederick Ruff and Mike O'Rourke.

All Aboard Hardware T&L

by Ron Fosner

Ron Fosner is the owner of Data Visualization, a company that specializes in creating and tuning high-performance, real-time software graphics systems for the consumer and professional markets. Ron specializes in real-time 3D graphics, code optimization, and hacks to get the maximum visual effect possible on consumer PC systems. He's the author of numerous articles on graphics and optimizations that have appeared in Dr. Dobb's Journal, Microsoft Systems Journal, *and* Game Developer. *He wrote the book on OpenGL programming for Windows as well as some of the OpenGL and Direct3D articles that appear in the Microsoft Developer Network CDs. His weapon of choice is a railgun, while his whiskey is Booker's. Send him your questions or suggestions at ron@directx.com, say hi at the GDC, or visit the Data Visualization web site (not in any conceivable way associated with Microsoft).*

If you've been working on any kind of game that takes advantage of a 3D graphics accelerator, you're probably painfully aware of the rapid pace at which 3D hardware evolves.

Just a few years ago most games didn't even consider using textures. Today of course, things are very different. On-board textures and texture state management are fully supported by the major graphics APIs, and most graphics cards come with gobs of texture memory. We also have the benefit of the AGP bus — a rapid data path designed exclusively for use by graphics cards. More recently, CPU manufacturers added specific instructions (3Dnow! and SIMD, for example) to pump out more vertex transformations and use fancier lighting and blending operations.

Advances such as these came about largely because graphics card manufacturers have attempted to meet the texture management and fill rate demands that games place upon the graphics API. Many of these advances were the result of porting features available on SGI workstations over to PC graphics cards, where consumer graphics chips manufacturers bottom-fed off of the SGI-developed hardware features, in order of easiest feature to hardest.

FIGURE 1. *The tree — gobs of textured, lit polygons rendered in real-time.*

As we've scarfed up more and more CPU cycles attempting to out-Carmack each other, we've been exercising those graphics APIs, pumping more and more textures and polygons through them, forcing the CPU to do more of the work. If your game isn't doing its own transforms and lighting operations (T&L), it's likely that the API driver is doing the setup and letting the CPU crunch those numbers.

Fortunately, we recently entered a new evolutionary phase in which T&L can be performed on 3D objects by a dedicated graphics geometry processor residing on the graphics card. A number of such graphics cards already exist, and more are coming out every quarter. Nvidia made the biggest splash with its GeForce 256 chip, which came out last fall and can be found on Creative Labs' 3D Blaster Annihilator, LeadTek's WinFast GeForce 256, and Elsa's Erazor X boards. S3's Savage 2000 chip, found on the Diamond Viper II board, also provides hardware T&L capabilities. And of course 3Dlabs has supported geometry acceleration for a while (no matter what Nvidia claims about being first), although 3Dlabs' cards (including the Oxygen GVX1) have been targeted primarily at the professional market. You can expect that the rest of the major players will be shipping (or at least announcing) T&L boards by the end of the year.

So we stand at the cusp of a new age of game graphics. Think about it: no longer do we have to perform multiple floating-point calculations on a vertex before sending it off to the API to be rendered. Instead, you simply hand off the vertices to a graphics card and this work is done for you, allowing your CPU cycles to be spent on something else. Performing T&L calculations in hardware lets you make more complex 3D models and scenes, enabling those freed-up CPU cycles to be used on other tasks (such as better AI).

To answer naysayers who complain about the visual quality of scenes generated by hardware T&L boards (or more particularly, by the lighting calculations performed by OpenGL and Direct3D), I concede the fact that coding your own lighting routines can provide a distinctly better look and offer more flexibility than the stock lighting calculations provided by most graphics APIs. Perhaps that's why hardware-accelerated T&L hasn't been on the graphics card feature wish list of many game developers thus far. But taken as a whole, hardware-accelerated T&L is great because it means developers can spend less time on the rendering pipeline portion of their games.

Talking about hardware T&L acceleration today is a bit like talking about how big a car's engine is when you can only drive it in a parking lot. What you really need is a wide open space to rev up the engine and run that sucker flat out. Unfortunately, there aren't yet any games that really stress the T&L engines. Nobody has designed a game with 20,000 to 50,000 polygons per frame simply because it would choke the non-T&L boards out there right now. Since we're already choking the CPU with our T&L calculations, we're left with little room to add more tasks when we already hog the machine. Hardware T&L enables us to load up the visual details of a scene while offloading a lot of the calculations involved onto the graphics geometry processor. While there's not much to actually using hardware T&L in your application (you just let the graphics API do the T&L), this article explains how to figure out if T&L is available through Direct3D (which has the only "standard" way of reporting if hardware T&L is available), and explores cubic environment mapping, a new feature that most hardware manufacturers have included with their T&L engines that simplifies the task of reflecting the environment around an object.

Leveraging Hardware T&L in Games

While writing this article, I tested an Nvidia GeForce 256–based card, which came with some demos. One of the more interesting demonstration applications generates a tree procedurally using settings for the number of branches, leaves, and so on, which is lit by eight fireflies (simulated by eight colored positional light sources). You can adjust the demo settings to make the tree as bushy as you'd like, so I cranked up the level of detail (LOD) to see what would happen (Figure 1). At "normal" LOD settings the tree was still quite bushy, looked good, and ran at a reasonable speed. But at the highest LOD setting, my 450MHz Pentium III workstation slowed down to about one frame per second. The reason for this can be seen in Figure 2 where you can see the high-LOD scene in wireframe mode. Notice the incredible amount of detail in that scene. They say a picture is worth a thousand words, and in this case it's worth literally tens of thousands of extra triangles and quads.

Create the Next Generation.

3D Studio MAX® The best games of today and tomorrow start with 3D Studio MAX. Built with a commitment to Open Source, and featuring expansive Level Design capabilities and innovative Bezier Surface tools, 3D Studio MAX is the production proven, future-worthy game solution.

discreet

Find out what's next. Visit our Game Zone at www.discreet.com

See us at the Game Developer Conference, Booth #1328

Autodesk and 3D Studio MAX are registered trademarks, and Discreet is a trademark of Autodesk, Inc. in the USA and/or other countries. All other brand names, product names or trademarks belong to their respective holders.
© Copyright 2000 Autodesk, Inc. All rights reserved.

You might wonder why I mention a 1FPS scene. Just look at the incredible detail in the scene shown in Figure 2. This is far beyond anything we'd consider putting in game scenes previously, and without hardware T&L, it wouldn't even be rendering that fast. But I assure you that frame rates for scenes such as this will quickly escalate. (Remember when we were all trying to figure out how to use texture mapping just a few short years ago? That problem was solved by the AGP bus and gobs of video RAM, and likewise I expect that today's T&L challenge will not be around for long.)

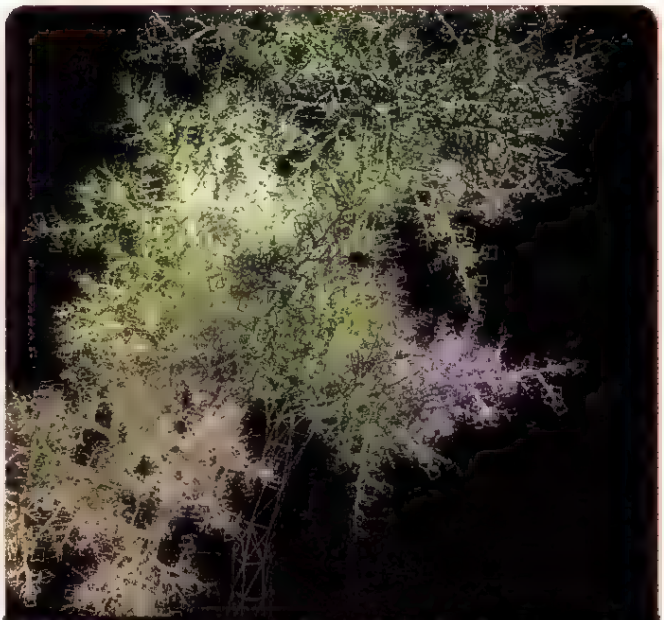


FIGURE 2. *The tree in wireframe illustrates the complexity of the rendering job.*

Taking advantage of hardware T&L means that you may have to change the way you do things. Above all, you have to let go of the idea that you can code T&L routines that perform faster than the same solution performed in hardware. To make sure you get T&L acceleration, two requirements must be met. First, you have to let the graphics API do the hardware transformation and lighting calculations for you, even if your engine already does them. (Yes, I know you finally got your quaternion code working, but it's time to move on to better things.) Second, you must verify that you have hardware T&L and then you need to enable it in the graphics API. If you're doing your own lighting or transforms (but not both), then you'll get some benefit from using hardware T&L. Thus, games such as UNREAL that do their own T&L won't see any benefit from hardware T&L accelerators. Games such as QUAKE 3: ARENA that do their own lighting calculations but let the graphics API do transformations for them will get some benefits from hardware T&L.

Today, only the two major graphics APIs, Direct3D and OpenGL, support hardware T&L. You must be using one of these two APIs in order to enable hardware T&L. I suspect that if 3dfx continues to push Glide, we'll see a T&L card and Glide driver from 3dfx in the near future. For now however, it's strictly an OpenGL and Direct3D show if you want hardware T&L acceleration.

For Direct3D to take advantage of 3D accelerators that support transformation and lighting operations in hardware, check for the `D3DDEVCAPS_HWTRANSFORMANDLIGHT` capability flag during the enumeration of devices. This flag is located within the `dwDevCaps` member of the associated `D3DDEVICEDESC7` structure and is returned when you call the `IDirect3D7::EnumDevices` method. Alternately, you can check the `GUID` of the device being enumerated — T&L devices will be identified with `IID_IDirect3DtnLHalDevice`. (Previously, we'd look for `IID_IDirect3DHALDevice` and stop there.) Note that on a T&L-supported card you'll find two hardware devices — the regular hardware HAL and the hardware T&L HAL, even though they may be the same device. (This is done for backwards compatibility reasons.)

To make sure you get T&L hardware acceleration in your game under Direct3D, you'll need to add a preference in your code for the T&L device over the regular HAL device. Remember you're talking to some third-party driver code here, so just because the card is in the machine doesn't mean it's automatically turned on. There seems to be a preference among the bleeding-edge chip makers towards 32-bit mode, so it might turn out that the new T&L interface only works on 32-bit color depths, while the legacy non-T&L interface is kept around for any 16-bit stuff. Remember that your mileage may vary depending on the graphics card and driver combination your game will use.

Using T&L hardware under OpenGL is a little different. Although the latest version of the OpenGL specification (version 1.2) doesn't directly mention support for hardware T&L, it doesn't really need to. With OpenGL, you just ask for a pixel format and you get "yes/no" information from the API about the availability of graphics acceleration. Since it is basically "yes" or "no" if you have hardware acceleration, and since in most cases folks let the API do their T&L (or perhaps just the transforms), you'll automatically benefit from code that was written before hardware T&L. By simply selecting the hardware-accelerated driver interface and letting the API do the T&L (or just transform *or* lighting), your program will automatically benefit. Gee, what a great API.

This is the spot in the article where I would have liked to insert a big chunk of code that you could cut and paste into your game and modify as necessary to get the best possible T&L acceleration. But hey, if you let the API do your T&L for you, then the code is exactly like you'd have written it if you weren't planning to implement T&L acceleration. In other words, implementing this support doesn't have to be difficult. The only tricky part is enumerating and selecting the T&L driver under Direct3D, since the T&L driver has been given a new `GUID`.

Drawbacks to Hardware T&L

Is there a downside to using hardware T&L? There can be. First, let me clear up one common misconception. Recently, people testing the GeForce 256 (one of the first consumer chips to support hardware T&L) have claimed that they cannot see much, if any, performance difference between using T&L acceleration and not using it. Some of this might be due to the fact that some of the tests being performed aren't real-

MAXIMUM SecuROM™ SAFETY

The dramatic growth of CD-ROM piracy world-wide is causing the loss of billions of dollars to the software industry. To counter this Sony DADC has developed SecuROM™, a copy protection solution fully integrated in your CD-ROM title. SecuROM™ uses an advanced electronic fingerprint to differentiate original CD-ROMs from pirated discs.

SecuROM™ offers you the following unique combination of advantages:

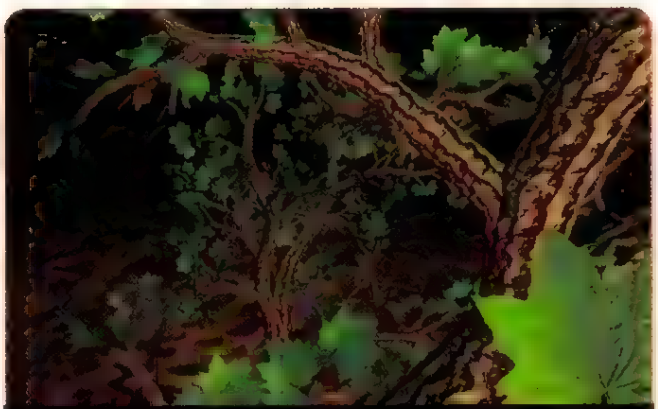
- SecuROM™ is the first and only system on the market **not using unreadable sectors (C2 errors)** on the disc to create an electronic fingerprint. Therefore, it offers the **best compatibility** with all CD-ROM and DVD-ROM drives. Your technical support helpline will appreciate the benefits of this.
- SecuROM™ offers a high security level that **prevents illegal consumer copying and professional piracy.** This will allow customers to experience the benefits of extended product shelf life, thereby maximizing profits through additional product sales'.
- **Affordable:** Currently publishers are investing huge amounts for both title development and marketing. Compared to this SecuROM™ is a **small investment** which easily will pay off.
- **Fast:** With the new SecuROM™ Online Encryption Toolkit you avoid any additional leadtime. This allows you to **encrypt your title via the Internet** and complete an inhouse testing of your Gold Master.
- The new **SecuROM™ Smart Card Protection** offers benefits like Pre-Release CD-R protection, Try & Buy solutions and Software on Demand business models. Also, with SecuROM™ no extra storage space is needed even for your most space intensive titles.

Sony Disc Manufacturing

Contact us for more Details
phone: 800-430-9665
e-mail: rhewitt@disc.sony.com
http://www.securom.com
http://www.sdm.sony.com

SecuROM™
Protecting your investments

Now available world-wide at Sony CD Manufacturing plants in USA, Europe, Mexico, Australia and Hong Kong as well as at other authorized independent CD-plants.



A close-up of a vertex-rich tree.

Vertex-rich tree in wireframe mode, showing its high polygon density.

ly designed to assess T&L. After all, if the difference between a T&L driver and a non-T&L driver is merely some capability bits, but OpenGL programs (which have always had built-in matrix memory and math routines) still let the hardware handle T&L, is it reasonable to assume that T&L won't occur in hardware if you're running on a T&L card? (Remember that the program doesn't have to change — it still passes the same matrix information and vertex list to the API.) To truly see the difference this new generation of hardware provides, we need new tests to stress the T&L capabilities of the chip — tests that churn through huge numbers of polygons per frame. You should be able to double or triple the number of polygons in a game compared with the numbers we're used to using, and see little performance impact on a T&L accelerator. Such tests should be available in short order — I'm creating some of these tests, and I'll post the results on my web site when they're finished. (If you have any suggestions or have done your own testing and would like to share the results, feel free to contact me.)

Aside from these misunderstood testing issues, there can be genuine instances when implementing hardware T&L will hurt application performance. Recall that computing T&L in hardware means that your game must give the card all of a scene's vertex data, as well as the lighting and translation parameters. This is similar in concept to sending texture data to the texture memory on the graphics card. It's faster to have the texture memory reside on the graphics card, but it's slower if you need to change or access that data. But there may be occasions when you need to get the results of a T&L calculation back over the AGP bus and back into system memory so you can use the results. For instance, your game might require the location of some object in worldspace, perhaps in order to calculate whether two objects collided. If you're using hardware T&L, it can be computationally expensive to request this information from the API since the API might be doing something else (caching data to optimize calculations, for example). To retrieve information from the hardware, the API must flush all pending calculations to update its state before it can return that information. As such, repeatedly requesting data during a rendering cycle can trash the caching/optimization scheme. I brought this issue up at a 3D graphics roundtable discussion at the 1999 Game Developers Conference, but the issue hasn't been resolved yet by the APIs. (Again, if you have any ideas, pass them along.)

Some method is needed to mark certain state transformations as "volatile," so the data associated with these transformations doesn't get stuffed into the bowels of the graphics CPU. Caching the data in the driver is one way to maintain access to it, but you'd get no benefit from hardware T&L in this case if you requested the data frequently. Currently, the only solution to this problem is to let the graphics API take care of the T&L, and for you to maintain your own copy of the translations (so don't throw away all your matrix and vector code just yet) and perform your collision detection (and so on) using a local copy of the transformation data.

Yes, this "solution" is ugly. After all, if I want the API to handle T&L, I want to unload all my T&L code, not keep a duplicate around. All I can say to rebut this is that the kinks with this technology are still being worked out. (It reminds me of the time when someone first explained to me how to do multi-pass texture effects. My response was, "You want me to render the whole scene three times every frame? You're kidding!" Now APIs have taken over that chore for the most part, and we're trying to figure out the best way to let the graphics chip handle some of the calculations while still letting us peek at the results as necessary.)

Cubic Environment Mapping

Cubic environment mapping is a feature supported by the latest revisions of OpenGL and Direct3D that has debuted as a hardware-supported feature in the latest crop of hardware accelerators. An environment map is essentially a texture map of a scene viewed from one spot. In the past, such environment maps were generated for a scene or taken with a 360-degree camera or one with a fish-eye lens. This usually gave you a pretty distorted texture, but since the texture was going to wrap around some shiny object for the purpose of creating a reflection, inaccurate environment maps typically sufficed (no one complained much except artists). Now that new graphics hardware can generate and process cubic environment maps, scene-accurate, real-time reflections are a reality.

Because finding the right content shouldn't be it's own project.

Are you still buying production media content the old-fashioned way? Don't waste time or money searching through old boring libraries or so called on-line resources. Join the iNoiz revolution and hear what all the Noiz is about!

Noiz.com The definitive resource for production media content on the planet. With millions of high quality professional content files available in all the popular editing formats like WAV, AIFF, MP3, G2 Targa, Tiff, JPEG, and many others, your flexibility and creativity is limitless. And with instant search preview, purchase and download or custom CD burn, your content is only a mouse click away.

With Noiz.com, choose from some of the worlds best production libraries, custom content, sound effects, licensed and buy-out music, graphics, animation, 3D models, stock photography and access some of the most creative minds and products in the world all at one place. iNoiz.com

Whether your project is a feature film, game development, video, broadcast advertising or interactive multimedia, finding the right content shouldn't be it's own project. Join the revolution today and experience what all the Noiz is about!

iNOIZ.com

Real content for the Real World

www.inoiz.com

Experience us at the Game Developers Conference and NAB !

The advantage of the cubic environment mapping technique is that when compared with traditional spherical texture map methods, it's easier to generate and use the reflected images. As such, cubic environment maps let you make more complex scenes, render a scene multiple times, and create dynamic, realistic reflections of it. Another plus is that you don't get a singularity at the poles, which is the case with spherical texture maps. You can generate the environment textures at run time and update the environment maps, updating the textures every frame to keep the reflected texture map accurate to the current scene. Even if your environment map is static, it's still easier to generate the environment map by rendering the actual reflected scene and saving it.

There are two steps to implementing cubic environment mapping. First, generate the texture map of the environment surrounding the object on which the map will be applied. Direct3D uses a special texture layout scheme just for cubic environment maps that's based upon an unfolded cube with ordered faces — hence the "cubic" term. OpenGL's solution is similar, although it doesn't order the cube faces. The second step is simply to load up the faces of the cube by rendering the scene from the center of the cube for each cube face and voilà, you have your cubic environment map. The conceptual layout of the textures can be seen in Figure 3.

To create a texture to use as a cubic environment map using DirectX 7, you call the `IDirectDraw7::CreateSurface` method. A cubic map is created as a series of attached surfaces (or "complex surfaces," in Direct3D parlance). There are three things to keep in mind when creating a cubic environment map: You cannot create the cube's surfaces individually, the dimensions you specify are the dimensions of an individual side, and each side of the cube must be both square and a power-of-two length (32×32, or 64×64, or 128×128 pixels, and so on). Typically you render this environment map onto a small reflective object so your cube dimensions can be small as well. That's good news. Because this is a multi-pass rendering technique, the cubic texture should probably be kept as small as possible to maintain good performance. To generate the optimal map, I suggest you render your scene in a window and make the window smaller and smaller until you start to lose fidelity. Then round up the dimensions of the texture to the next power of two.

If your scene requires Z-buffering, you can create a separate Z-buffer and attach it to the cubic environment map. Alternately, you can detach the Z-buffer from your back buffer (assuming you created one) and attach it to the cube map before you render, then reattach it when you're rendering the scene "for real."

Rendering the cubic environment map to an object can get nasty. First, you allocate the environment map texture memory for the current scene, as shown in Listing 1. Assuming that we can call some "render" method on our scene to render all objects in their correct positions in worldspace, we just need to set the viewpoint to the center of the cubic environment map and render the scene into the map. You do this for each of the six sides of the cube. This gives us six additional renderings of our scene for each cubic-environment-mapped object within the scene. So far, so good. But things can get tricky when you have multiple cube maps that are within sight of

LISTING 1. *Cubic environment map code under Direct3D. This shows some Direct3D (DirectX 7 or better) code that will allocate a cubic environment map.*

```
// Assume pDD7 is a pointer to an IDirectDraw7 interface.
DDSURFACEDESC2 ddsd;
ZeroMemory((LPVOID)&ddsd, sizeof(DDSURFACEDESC2));
ddsd.dwSize = sizeof(DDSURFACEDESC2);
ddsd.dwFlags = DDSD_CAPS | DDSD_WIDTH | DDSD_HEIGHT | DDSD_PIXELFORMAT;
ddsd.ddsCaps.dwCaps = DDSCAPS_TEXTURE;

// Set the pixel format to a valid texture format here.

// Dimensions must be the same and must be a power-of-two.
ddsd.dwWidth  = ddsd.dwHeight = 128;

// Set caps for a system memory cube-map texture that is a valid render target surface.
// We set the DDSCAPS_3DDEVICE bit only if we want to use the environment map as a render
// target. If you're going to just load textures, you don't need this flag set.
ddsd.ddsCaps.dwCaps = DDSCAPS_COMPLEX | DDSCAPS_3DDEVICE | DDSCAPS_TEXTURE;

// Here's where we state that it's a cubic environment map and that we want all 6 faces
// created at once.

ddsd.ddsCaps.dwCaps2 = DDSCAPS2_CUBEMAP | DDSCAPS2_CUBEMAP_ALLFACES;
LPDIRECTDRAWSURFACE7 pddsCubeMap;

// And here's where we create the cube map complex surface.
if( FAILED( pDD7->CreateSurface( &ddsd, pddsCubeMap, NULL ) ) )
{
   // something bad happened.
}
```

FIGURE 3. *Conceptual layout of the cubic environment map.*

artists **animators** **game developers**

When your reputation depends on your performance

and reliablity

and graphics power

and interactivity

– our performance is your art.

hp®

invent

HP VISUALIZE WORKSTATIONS

visit: www.hp.com/info/dcc3

intel inside

pentium® III

Your reputation is built on your 3D animation, visual effects or games. And HP performance and reliability are exactly what you need because an HP VISUALIZE workstation is body armor for your reputation.

HP's VISUALIZE *fx*® graphics deliver the extreme performance you need to work interactively with million+ polygon models. Great graphics, the latest Intel® Pentium® III processors, HP-engineered reliability, and at a price you can afford.

Our performance is your art.

Intel, the Intel Inside logo, and Pentium are registered trademarks of the Intel Corporation.
Images (from left to right) courtesy of Alias|Wavefront, Major Damage™, ©2000 Chris Bailey and EPI/Ron Dicker.

each other. You run into the problem of accurately generating the reflections of the reflections, which must be repeated *n* times (where *n* is the number of times you want the reflected scene reflected in your environment map). A common solution to this problem is to use the previous frame's rendering in the reflection repeatedly for each frame, as long as you don't mind your reflections being off a frame in the reflections. It's not a perfect solution, but it still looks good most of the time. (Yet I doubt we'll see many games with multiple real-time cubic-environment-mapped objects anytime soon.)

Listing 2 illustrates the process for rendering a cubic environment map in pseudocode. This code describes the process within Direct3D; OpenGL supports cube mapping through an extension mechanism. (On my GeForce card it's the `EXT_texture_cube_map` extension.)

After you've reset the viewport and all matrices (in step 7), then you can render your scene again using the environment map we just created. The result is a texture map that looks like Figure 4.

If you are wondering how the texture coordinates are used (since the cube texture maps are treated as a 3D texture), let me explain briefly. The vertex texture coordinates are treated like a directional vector with an origin at the center of the cube. The largest coordinate axis is the one that selects the cube face. The remaining minor coordinate axes are divided by the larger, and these two values are treated as the 2D U and V coordinates to the cube face. From that point onward, the cubic environment map is treated just like a regular 2D texture map.

While the effects created by cubic environment maps are pretty neat, it does affect rendering speed. Without the benefit of hardware acceleration, a typical scene is rendered about two to ten times slower than without an environment-mapped object — hence the requirement for a T&L graphics card (which usually includes cubic environment mapping in hardware). But you do get spectacular effects out of it, as shown in Figure 5. This image shows a sphere that was environment-mapped with the texture from Figure 4. A more interesting example is shown in Figure 6, where there is subtle interaction between the environment and the model's hair: as the woman moves her head, the highlights on her hair change to reflect the environment.

While cubic environment mapping is a useful feature, I'd rather see it pushed further into the API. Right now it's got the feel of a hack, so hopefully as the API folks get more feedback they'll refine the API to make the setup and rendering process less onerous. While it's nice to have non-90-degree viewpoints, or to create the faces separately, 99 percent of the time you use the same defaults, so a lot of setup could be avoided by providing an easier-to-use set of default interfaces, thereby putting less of a setup burden on the programmer.

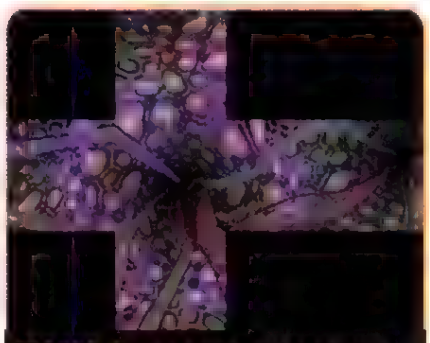


FIGURE 4. *The actual cubic environment map for the scene.*

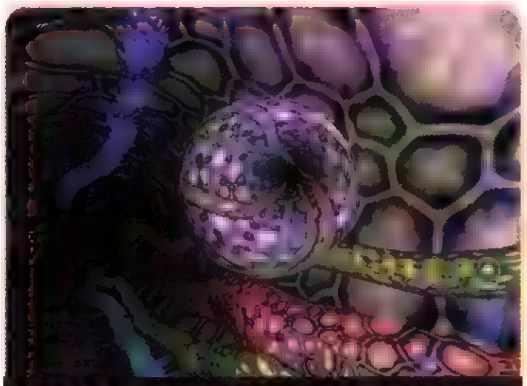


FIGURE 5. *The scene with a reflective sphere incorporating the cube map.*

LISTING 2. *Process of generating and rendering a cubic environment map in pseudocode.*

```
1. Prior to rendering the scene, render the cube map.
2. If the cube map dimensions are small, enable antialiasing for the cube rendering.
3. Set the viewport to the cube dimensions.
4. Set the perspective matrix to a 90-degree field of view for both the X and Y axes.
5. For all six faces of the cube map:
    a. Set the render target to the current face.
    b. Optionally swap the Z-buffer to this map.
    c. Set the viewpoint LOOK and UP vectors as follows for the particular pass:
        Cube face    LOOK Direction    UP Direction
        0            positive X        positive Y
        1            negative X        positive Y
        2            positive Y        negative Z
        3            negative Y        positive Z
        4            positive Z        positive Y
        5            negative Z        positive Y
    e. Set the modelview matrix according to LOOK and UP vectors centered at
       cube center.
    f. Begin scene.
    g. Render everything in the scene but this object.
    h. End scene.
6. Restore render target.
7. Reset viewport and all matrices.
```

2.01: A New Game Cycle

We're about to be handed a big dividend in the form of low-cost T&L calculations, and like it or not, if you're coming out with a 3D game for Christmas 2001, you ought to incorporate hardware T&L into your title if you want your game to be commercially competitive.

This is an issue that I think will have a revolutionary effect on the games market, but it will take a while for everyone to get comfortable with how best to incorporate T&L into the programming. The upside is that in the next few years we can expect to see some incredibly detailed games with spectacular environments. We can use this functionality to incorporate more complex models, better AI, or some other feature that would previously have placed too much strain on an already-burdened CPU.

For the time being, it's up to us to test out these new hardware features and do our best to give feedback to the chip manufacturers and API architects. In the meantime, we must help them in order for them to help us add complexity to our content. Test out one of the new T&L cards and see how you can get more content into your game.

FIGURE 6. *Example showing the subtle interaction between the environment lighting and the model.*

FOR FURTHER INFO

Hardware information and reviews:

Hard OCP
http://www.hardocp.com

AnandTech
http://www.anandtech.com

Tom's Hardware

Chip manufacturers supporting T&L:

Nvidia Corp.
http://www.nvidia.com

S3 Inc.
http://www.s3.com

3Dlabs Inc.
http://www.3dlabs.com

If You Can Make It, We Can Make It Talk.

Has the thought of lip-synching your character by hand got you down? No matter what your platform, skill level or project, LIPSinc™ can relieve you of this time-consuming task. Are you a Poser® fan? LIPSinc Mimic™ will add life to your characters. 3D Studio Max® user? Check out our plug-in LIPSinc Ventriloquist® *(available soon for Maya™ and SoftImage®)*. For large projects, you'll need LIPSinc Echo™ available as a batch processor or library. Add talking characters to any project. Achieve more accuracy, realism and expression than ever before—automated lip-synching from LIPSinc.

Eval versions of all LIPSinc products available on the website at

Going to the Game Developer Conference? See for yourself at booth #1118

How to Simulate a Ponytail, Part 2

by Chris Hecker

When I was a kid, I used to leave my room — or wherever I went, really — a total mess. I'd like to be able to say that I've changed and am a really tidy and responsible person now, but that would be a stretch. I am, however, going to clean up the awful mess I left us in after the last issue, where we had a bunch of equations, a whole lot of terms, and not much of a clue about what to do to get a ponytail simulator out the other end.

Swinging ponytail screenshot from the sample application used in last month's article.

When We Last Left Our Heroes...

There's no escaping the fact that you need to read last month's part one article ("How to Simulate a Ponytail," March 2000) in order to read this part two. I can't review it in any meaningful way, so I'm just going to set up our initial conditions from the end of last month's article and move on. Figure 1 shows the bodies and notation we're using, and Table 1 contains the equations we ended up with.

In part one, we decided to do the derivation for two constrained bodies first, to keep things manageable, and then later generalize it to the longer chain of bodies that make up the ponytail. At the end of part one, we had written out equations for the linear and angular accelerations of our simple two-body system when they were affected by the constraint force, f_c, as you can see in Equations 1 and 2. We also determined that Equation 3 was going to be the constraint equation we would attempt to satisfy at all times during the simulation using the constraint force. If we could satisfy Equation

Chris Hecker (checker@d6.com) is the Editor-at-Large of Game Developer.

3, and our simulation started with the position and velocity constraints satisfied, we'd have a constrained rigid body simulator. Finally, I said the end product of all the plugging and chugging with equations would be a linear system of equations looking like Equation 4: $\mathbf{A}f_c = b$. We'd solve this equation for the force of constraint, and then apply it back to the objects to stick them together.

Plug 'n' Chug

We originally derived Equation 3 because we needed the constraint equation to be in terms of accelerations rather than positions or velocities. Now that we've got it in acceleration space, we can enforce it with f_c, since we know forces can directly affect accelerations. Still, it's not immediately obvious how to get our f_c into Equation 3, where it can do some good.

Equation 3 is too abstract for our needs. It simply says the acceleration of the two constraint endpoints must be equal. It makes sense that the force of constraint, f_c, can affect the accelerations of the endpoints by pushing and pulling on the bodies, but how do we show this mathematically? First, we need to express the endpoint accelerations in terms of the body's linear and angular accelerations, which we know are directly affected by f_c via Equations 1 and 2.

Remember from part one (or from my original physics articles from *Game Developer*, referenced at the end of this article) that the equation for the acceleration of a point fixed on a rigid body — say, Body A — looks like this:

$$\ddot{p}_A = \ddot{R}_A + \alpha_A \times r_A + \omega_A \times (\omega_A \times r_A) \qquad \text{Eq. 5}$$

Equation 5 contains the second derivative of R_A (the vector to the center of mass of the body) and α_A (the angular acceleration of the body). These quantities are definitely affected by f_c as shown in Equations 1 and 2.

If we substitute Equation 5 and its counterpart for Body B into Equation 3, we get a very long equation. Then, if we substitute Equations 1 and 2 and their counterparts for Body B into the very long equation, we get an *extremely* long equation. At that point, our extremely long equation is in terms of our only unknown, f_c, and we can munge it around until we get something that looks like Equation 4. We could do this, but we'd probably go insane trying to keep all the terms straight with all their subscripts and whatnot, and I know I'd go insane trying to type all the intermediate stages into the evil Equation Editor.

We'll take a step back, and just work with Equation 5 for a little while. We can move ahead under the assumption that anything we do to Equation 5, we can do to its Body B partner. If we can simplify Equation 5 before substituting it into Equation 3, then we can do the same for the B version and we'll stay sane.

One Term at a Time

Look at the first term on the right hand side of Equation 5, the acceleration of R_A. Equation 1 just drops into Equation 5 in place of this term, and we get Equation 6:

$$\ddot{p}_A = M_A^{-1} f_c + M_A^{-1} F_{EA} + \alpha_A \times r_A + \omega_A \times (\omega_A \times r_A) \qquad \text{Eq. 6}$$

This is already starting to get messy. We can simplify a bit by introducing a new term, b_A. We'll use b_A to hold all of the "known" terms in the equation. The known terms are those that contain quantities whose values we know how to calculate at any given time. So, as we discussed in part one, the external forces are all known at a given timestep, meaning we can stuff the F_{EA} term into b_A. Also, the last term in Equation 6 is known because it only contains angular velocities and the position vector, r_A, both of which are known at any timestep since they were integrated forward from a previous

FIGURE 1. *Bodies and notation from last month's article.*

TABLE 1. *Review equations from last month's article.*

ACCELERATION EQUATIONS. Equations 1 and 2 are the linear and angular acceleration equations for the rigid body A in terms of the external forces and torques (denoted with a *E* subscript) and the force of constraint, f_C. Body B's equations would be the same with *B* subscripts and a $-f_C$ in place of the f_C terms because the constraint force is applied negatively to Body B.

$$R_A = M_A^{-1} f_C + M_A^{-1} F_{EA} \qquad \text{Eq. 1}$$

$$\alpha_A = I_A^{-1}(r_A \times f_c) + I_A^{-1}\tau_{EA} - I_A^{-1}(\omega_A \times L_A) \qquad \text{Eq. 2}$$

CONSTRAINT EQUATION. Equation 3 is the second derivative of the position constraint equation, $p_A - p_B = 0$. This equation specifies that the positions (and velocities and accelerations) of the endpoints of the constraint vectors must be equal at all times.

$$\ddot{p}_A - \ddot{p}_B = 0 \qquad \text{Eq. 3}$$

OUR GOAL. At the end of this article, we'd better have a system of linear equations that looks like Equation 4.

$$\mathbf{A}f_c = b \qquad \text{Eq. 4}$$

timestep. So, our b_A looks like this so far:

$$b_A = M_A^{-1}F_{EA} + \omega_A \times (\omega_A \times r_A) \quad \text{Eq. 7}$$

And, our simplified Equation 6 looks like this:

$$\ddot{p}_A = M_A^{-1}f_c + \alpha_A \times r_A + b_A \quad \text{Eq. 8}$$

The substitution of Equation 2 for α_A is more complicated. First, the α_A is inside a cross product, which means the entire right-hand side of Equation 2 is going to have to go into the first term of that cross product. This makes perfect sense mathematically: α_A is a vector, and so the right-hand side of Equation 2 is a vector as well — it's simply composed of a bunch of other vectors. It's a big mess symbolically, though, because replacing the single symbol on the left-hand side of Equation 2 with the multi-term expression on the right-hand side makes the cross product pretty much unreadable, and we have to use parentheses to keep everything straight.

We'll concentrate solely on the α_A term in Equation 8 and ignore the other terms for a moment. We substitute in Equation 2 and use the fact that the cross product distributes across addition and subtraction:

$$\alpha_A \times r_A - \left[I_A^{-1}(r_a \times f_c)\right] \times r_A + \left[I_A^{-1}\tau_{EA}\right] \times r_A - \left[I_A^{-1}(\omega_A \times L_A)\right] \times r_A \quad \text{Eq. 9}$$

Now, before dropping Equation 9 back into Equation 8, let's try to stick a bunch of it into b_A to get it out of the way. The first term on the right-hand side of Equation 9 contains f_c, so we need to keep it around, but the other two terms are both known, since they contain only external torques, velocities, momenta, and positions. Away into b_A they go, leaving us with:

$$b_A = M_A^{-1}F_{EA} + \omega_A \times (\omega_A \times r_A) + \left[I_A^{-1}\tau_{EA} - I_A^{-1}(\omega_A \times L_A)\right] \times r_A \quad \text{Eq. 10}$$

I "un-distributed" the cross product of the two known terms in Equation 9 when writing Equation 10 to make it a bit shorter.

Our equation for the constraint endpoint acceleration now looks like this:

$$\ddot{p}_A = M_A^{-1}f_c + \left[I_A^{-1}(r_A \times f_c)\right] \times r_A + b_A \quad \text{Eq. 11}$$

As you can see, it's much simpler than it could have been, but we've still got some work to do.

A Breather

Let's take a break and assess our situation. We have Equation 11, which is an equation for the acceleration of Body A's constraint endpoint in terms of the known quantities (most of which are tucked away in b_A), and the unknown constraint force, f_c. That is, at any given time we can calculate the value of the b_A vector, and plug it into Equation 11. We can also calculate all of the other known terms on the right hand side of Equation 11, such as the mass, inertia tensor, and constraint vector, r_A. The exception is f_c; we don't know it in advance. In fact, our whole goal is to solve for f_c so we can plug it back into Equations 1 and 2 to find the acceleration of the bodies under constraint.

Since f_c is our unknown, we need to get it in a better position to be manipulated. The first term on the right-hand side of Equation 11 is pretty reasonable, since it's just a matrix times f_c. This term looks a bit like Equation 4, so we know we're getting close. However, f_c is stuck inside two cross products in the second term, which is a far cry from Equation 4.

Cross Products

How do we get f_c out of the cross products? Cross products are notoriously hard to manipulate, unless you have the following definitions in your bag of tricks:

$$a \times b = -b \times a \quad \text{Eq. 12}$$

$$a \times b = \tilde{a}b = \begin{vmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{vmatrix} \begin{vmatrix} b_1 \\ b_2 \\ b_3 \end{vmatrix} \quad \text{Eq. 13}$$

Equation 12 is the familiar rule stating that when you reverse the cross product terms, the resulting vector is negated. This is what you see when you accidentally compute a triangle's normal by crossing the edges in the wrong direction — you get the inverted normal.

Equation 13 defines the "tilde operator," which, when applied to a vector, creates the matrix shown in the equation. It just so happens that this tilde-matrix of vector a — also called the "skew symmetric matrix of a" — times vector b gives the same resulting vector as taking the cross product of a and b (multiply the matrix-vector product on a piece of paper to double-check it for yourself). This is a great trick, because it turns a cross product into a matrix-vector product, which we know how to manipulate.

Now we can pound on our cross product term. First, let's get f_c on the right side of the expression by applying Equation 12:

$$\left[I_A^{-1}(r_A \times f_c)\right] \times r_A = -r_A \times \left[I_A^{-1}(r_A \times f_c)\right]$$

Next, we "tilde-ize" the outer cross product:

$$-r_A \times \left[I_A^{-1}(r_A \times f_c)\right] = \tilde{r}_A I_A^{-1}(r_A \times f_C)$$

Notice how the outer brackets aren't needed anymore, because now we just have a matrix multiply of the tilde'd r_A and the inverse inertia tensor. Finally, let's tilde-ize the inner cross product:

$$-\tilde{r}_A I_A^{-1}(r_A \times f_c) = -\tilde{r}_A I_A^{-1}\tilde{r}_A f_c$$

Our result is simply three matrices times our unknown vector. Let's put this result back into Equation 11 and group the terms to isolate f_c:

$$\ddot{p}_A = M_A^{-1}f_c - \tilde{r}_A I_A^{-1}\tilde{r}_A f_c + b_A$$

$$\ddot{p}_A = \left[M_A^{-1} - \tilde{r}_A I_A^{-1}\tilde{r}_A\right]f_c + b_A$$

$$\ddot{p}_A = \mathbf{A}_A f_c + b_A \quad \text{Eq. 14}$$

Now we're in business. I've renamed the grouped matrices "$\mathbf{A}_A$" to highlight the structure of the equations. We have a

Are you cashing in on your online games?

E-lysium solution allows game developers and content providers to reach new demographics not accessible to credit cards and serves as an alternative payment method for amounts ranging from just a few cents to hundreds of dollars.

Game developers can take advantage of the newest software solution to make their on-line games available for payment over the Internet. The new Game Developers Kit (GDK) from e-lysium enables developers to adapt on-line games to accept Internet cash for payment. You can charge your players based on the elapsed time, number of users or by any custom defined preference. It's just a drag and drop away. Start accepting Internet Cash today and get paid for your online games.

From interactive games to digitally delivered information from downloading music on-line to chat and advisory services, e-lysium provides the technology for cash payments over the Internet.

e-lysium

Visit our site at GAME Developers Conference 2000, Booth 1528
March 8-12, 2000 San Jose Convention Center, San Jose, CA

©2000 E-lysium Transaction Systems Inc

E-lysium Transaction Systems Inc., 4601 Sheridan Street, Suite 500, Hollywood, FL 33021
Voice 954 987 6900 Facsimile 954 987 6590 Web site **www.e-lysium.com**

known matrix, $\mathbf{A}_A$, and a known vector, b_A, and our unknown f_c. We're not quite at Equation 4, but we're extremely close. We only need to get rid of the p_A acceleration term, and that's our cue to substitute back into Equation 3.

Body B

If you look at all the work we did to get from Equation 5 to Equation 14, you'll see that very little of it depended on whether we were talking about Body A or Body B. There's really just one difference between the bodies, and we've already mentioned it: f_c is positive for Body A, but negative for Body B. This means we can simply rewrite all the equations with B subscripts, and if we're careful to substitute in $-f_c$ wherever f_c shows up, we'll have valid equations for Body B. We don't have to rederive everything.

We can actually just write the Equation 14 for Body B by inspection:

$$\ddot{p}_B = \mathbf{A}_B(-f_c) + b_B = -\mathbf{A}_B f_c + b_B$$

Eq. 15

The $\mathbf{A}_B$ matrix and the b_B vector are calculated exactly as shown above for Body A, and we simply negate the f_c term. Equation 3 tells us we can subtract Equation 15 from Equation 14 to get 0, assuming we're enforcing our constraint properly. Let's write the subtraction, taking care to get our signs right:

$$\ddot{p}_A - \ddot{p}_B = \mathbf{A}_A f_c + b_A + \mathbf{A}_B f_c - b_B = 0$$

Finally, let's group our terms to match Equation 3:

$$[\mathbf{A}_A + \mathbf{A}_B] f_c = -b_A + b_B$$

Eq. 16

That's it. We have a linear system matching Equation 4, where $\mathbf{A} = \mathbf{A}_A + \mathbf{A}_B$ and $b = -b_A + b_B$. $\mathbf{A}$ and b are known, and f_c is unknown, meaning that at any given time, we can construct Equation 16 for the two rigid bodies, and then solve it for f_c to find the constraint force that will hold the bodies together.

The Simulation Algorithm

Now we're ready to outline the overall simulation algorithm for two rigid bodies with one constraint:

1. Compute the external forces: F_{EA}, τ_{EA}, F_{EB}, τ_{EB}.
2. Compute the $\mathbf{A}$ matrix and the b vector.
3. Solve the linear system for f_c.
4. Apply f_c and the external forces to the objects.
5. Integrate forward.

After step three, f_c is a known force, just like the external forces. The forces can now all be applied to the bodies in the usual way. Remember, f_c is applied at the tip of the constraint vector, so it will induce torque on the objects as well as apply a force to the center of mass. Also remember that f_c is applied negatively to Body B.

The Linear System

So, how do we solve the $\mathbf{A}f_c = b$ linear system? This is actually the easiest part of the algorithm in some sense, because there are so many different ways to do it. Solving linear systems on computers is the most studied area of numerical analysis, and there are hundreds of books about doing it right and lots of free source code. You could probably write your own Gaussian Elimination routine in a few lines of C, or you could download some fancy numerical linear algebra package if you're so inclined. If you're just interested in solving the 3×3 system in Equation 16, you could even use Cramer's Rule or just solve the system by hand symbolically, but those techniques won't scale to the larger systems that occur when we add bodies. When I wrote the sample application for this article, I downloaded a simple linear system solver from the web (http://www.netlib.org) and hacked it into my program. See the references for details on the sample application.

Kinematic Control

Before we generalize our derivation to multiple bodies, let's talk about how kinematic control fits into our formulation. From last month, you'll remember the head of the character is kinematically controlled, meaning its movements are already known from an animation. It's easy to integrate animated bodies into our algorithm.

First, we need to be able to generate values for the known position, velocity, and acceleration of the kinematically controlled body at a given point in time. We can find the equations for these values from our animation system in most cases. The position equation is simplest — we have to have that around if we're animating the body in the first place. The velocity and acceleration equations are attained by differentiating the equation for position. If we're interpolating keyframes, then the interpolation function will give us the velocity at a given time when we differentiate it. Another differentiation gives us the acceleration. For example, if we're linearly interpolating positions between keyframes, the velocity will be constant and the acceleration will be zero. Linear interpolation is not continuous at the keyframes themselves because the direction changes sharply, so be careful about differentiating in those areas. If we're linearly interpolating joint angle keyframes, our velocities and accelerations will be nonlinear, but still derivable from the equations. If we're doing a more sophisticated continuous spline interpolation, our derivatives will be even more complicated, but we should still be able to attain equations for the velocity and acceleration

Once we've gotten the linear and angular positions, velocities, and accelerations from our animation system, we use these known values everywhere they appear in our equations. Equations 1 and 2 for the kinematically controlled body become known values, rather than equations depending on the force of constraint. All of the animating body's kinematic quantities are now known and end up in

dynamics toolkit 2.0

If you've looked at using dynamics middleware before, you'll find our latest offering faster, more stable and powerful enough to really let you make a difference to your game.

If you havn't considered it before, we'd like to show you what we can do.
You'll find our Dynamics Toolkit is designed to mesh with your game design not dictate it.

We can give you a physics solution that works out of the box or can be tailor made for your title. Most components are supplied with full source code and open pricing and all are available on PS2 and PC
(Windows or Linux).

Come and talk to our application engineers at GDC Booth 1708 and be one of the first to see it in action or visit **www.mathengine.com**

mathengine

the b vector. When we compute f_c, the computed force will make the dynamically controlled body obey the movement of the constraint due to the kinematically controlled body's animation. We don't apply the constraint force to the animating body because, well, it's animated, not simulated.

Multiple Constraints

Two bodies does not a ponytail make. Do we have to rederive everything when we want to do three bodies with two constraints in a chain, not to mention N bodies with $N-1$ constraints? Thankfully, the answer is no. The equations for multiple bodies and constraints are very similar to those we've already derived, but we need to talk a bit more about the structure of the multi-constraint problem before we can extend them.

The first thing to notice is that we have to solve for all of the constraint forces simultaneously. If I have a chain of three bodies, and I pull up on the top body, not only does the middle body have to feel the yank, but the bottom body does as well. If we didn't solve simultaneously, the force of the pull would ripple down the chain in the order we solved the constraints, and the chain would separate. This is not the behavior we want.

Because we need to solve simultaneously, all of the constraints need to be represented in the equations we write. This forces us to develop some new notation that will scale to multiple constraints. Bodies can now have two constraints attached to them, rather than just having one as in our two-body derivation. It turns out that it makes the most sense to be "constraint-centric" in our notation, numbering the constraints and having them refer to the bodies rather than having the bodies refer to the constraints.

Figure 2 shows this notation. The constraints are numbered 1, 2, and 3; if we were being completely general we'd call them $i-1$, i, and $i+1$. We're going to talk about the middle joint, number 2. We'll call the forces at each constraint f_{c1}, f_{c2}, and f_{c3}. The constraints attach the two bodies on either side of the joint, and I've chosen the subscripts u and d to stand for the "upstairs" body and the "downstairs" body relative to the joint in the figure. So, the body between joints 1 and 2 is the upstairs body of joint 2, and the other body is the downstairs body of joint 2. The constraint endpoint of the top body for joint 2 is denoted p_{2u}, and the endpoint from the bottom body is p_{2d}, and so on. The constraint endpoints and the r vectors are still attached to their respective bodies, but they're numbered relative to the joints. Notice that the u-body of joint 2 is the d-body of joint 1.

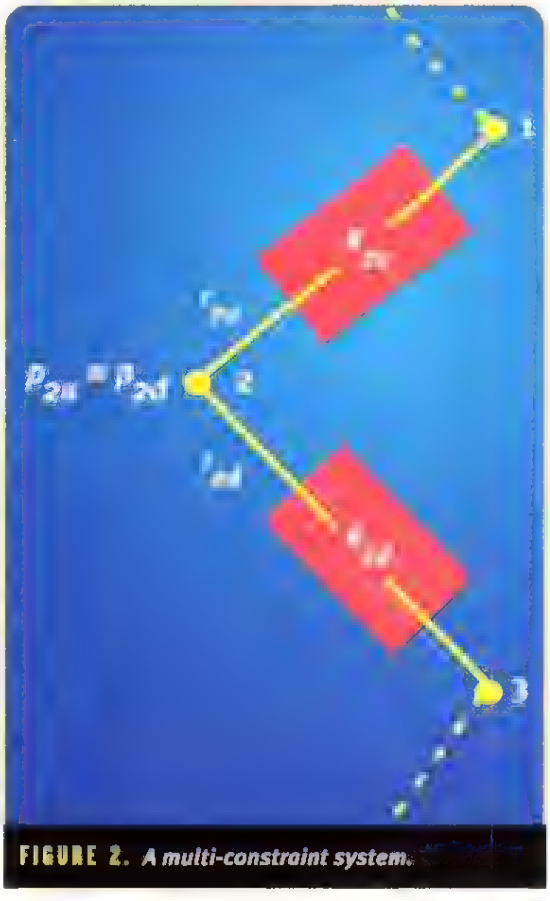


FIGURE 2. *A multi-constraint system.*

While I make no claims to the elegance of this notation, it will let us get the job done.

General Acceleration Equations

Given the new notation, we could rederive all of our equations for the new general constraint. We don't have the space for that (and it's almost identical to our previous derivation), so we're going to skip ahead and show the structure of the equations we end up with for joint 2. The accelerations of the two endpoints associated with joint 2 look like this:

$$\ddot{p}_{2u} = \mathbf{A}_{2u1} f_{c1} + \mathbf{A}_{2u2} f_{c2} + b_{2u} \qquad \text{Eq. 17}$$

$$\ddot{p}_{2d} = \mathbf{A}_{2d2} f_{c2} + \mathbf{A}_{2d3} f_{c3} + b_{2d} \qquad \text{Eq. 18}$$

A single joint is affected not only by its own constraint force, but also by the constraint forces on either side of it. This is because the body accelerations are modified by all the constraint forces acting on them, and those body accelerations appear in the equation for the constraint. Put another way, the top body's motion at joint 2 is dependent on what joint 1's force is doing, in addition to what joint 2's force is doing.

I haven't described what the $\mathbf{A}$ matrices look like exactly, but they'll be very similar in composition to the $\mathbf{A}$ matrices we derived above, so we can just deal with them symbolically here. They're subscripted to describe their function: $\mathbf{A}_{2u1}$ is the matrix for joint 2's upstairs body that multiplies f_{c1}. In English, $\mathbf{A}_{2u1}$ describes the acceleration effect f_{c1} has on joint 2's upstairs endpoint. Put yet another way, $\mathbf{A}_{2u1}$ maps the force from joint 1 to an acceleration at joint 2, through the body. To belabor the point a bit more, if you look at the expression that makes up $\mathbf{A}_{2u1}$ (once you've derived it, of course!), you'll see that it converts f_{c1} to accelerations on the center of mass, and then maps those accelerations out to p_{2u}.

Although I didn't mention this way of thinking above, the original $\mathbf{A}_A$ and $\mathbf{A}_B$ matrices from the two-body derivation work the same way. The linear acceleration is transferred through the M^{-1} term, and the angular acceleration is transferred through the cross product (or tilde matrix) and inertia tensor term. In $\mathbf{A}_A$ and $\mathbf{A}_B$ we're mapping the constraint force from the joint, down through the body, and back up to the same joint, but the

Visual C++ Developers:

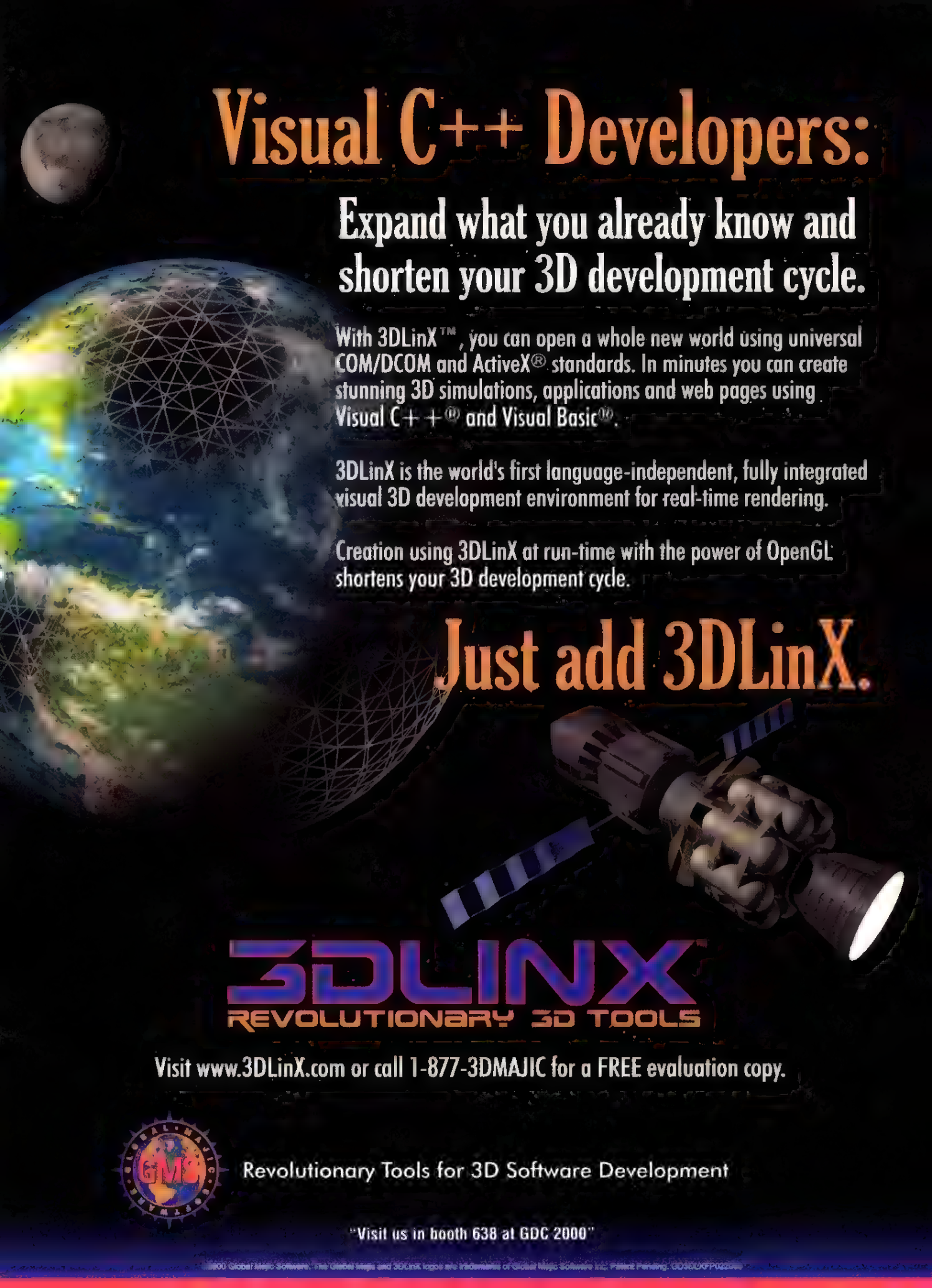
Expand what you already know and shorten your 3D development cycle.

With 3DLinX™, you can open a whole new world using universal COM/DCOM and ActiveX® standards. In minutes you can create stunning 3D simulations, applications and web pages using Visual C++® and Visual Basic®.

3DLinX is the world's first language-independent, fully integrated visual 3D development environment for real-time rendering.

Creation using 3DLinX at run-time with the power of OpenGL shortens your 3D development cycle.

Just add 3DLinX.



3DLINX
REVOLUTIONARY 3D TOOLS

Visit www.3DLinX.com or call 1-877-3DMAJIC for a FREE evaluation copy.

Revolutionary Tools for 3D Software Development

"Visit us in booth 638 at GDC 2000"

2000 Global Majic Software. The Global Majic and 3DLinX logos are trademarks of Global Majic Software Inc. Patent Pending. GD3DLXFP02000

principle still applies. In Equations 17 and 18, $\mathbf{A}_{2u2}$ and $\mathbf{A}_{2d2}$ are similar to $\mathbf{A}_A$ and $\mathbf{A}_B$, since they map f_{c2} to acceleration back at joint 2.

We've also adopted the convention of applying the constraint force positively to the *u*-body and negatively to the *d*-body, which accounts for the negative signs in Equations 17 and 18.

The Multi-Constraint System

If we subtract Equation 18 from Equation 17 and group terms, we get the constraint equation we must satisfy for joint 2:

$$-\mathbf{A}_{2u1}f_{c1} + [\mathbf{A}_{2u2} + \mathbf{A}_{2d2}]f_{c2} - \mathbf{A}_{2d3}f_{c3} = -b_{2u} + b_{2d} \qquad \text{Eq. 19}$$

This equation has the three unknown vectors in it, but it's only one vector equation. To solve a linear system, we need as many equations as we have unknowns. Where will we find the other equations? From the other constraints, naturally.

Let's assume for the moment that the system in Figure 2 has four bodies and the three constraints shown. In other words, although they're only hinted at in the figure, there is a body above joint 1 and a body below joint 3. These outlying bodies each has only one constraint (joint 1 for the upper body and joint 3 for the lower body, obviously), and they are the endpoints of the chain in this example. Given this system, the equation for joint 1 is:

$$[\mathbf{A}_{1u1} + \mathbf{A}_{1d1}]f_{c1} - \mathbf{A}_{1d2}f_{c2} = -b_{1u} + b_{1d} \qquad \text{Eq. 20}$$

And the equation for joint 3 is:

$$-\mathbf{A}_{3u2}f_{c2} + [\mathbf{A}_{3u3} + \mathbf{A}_{3d3}]f_{c3} = -b_{3u} + b_{3d} \qquad \text{Eq. 21}$$

Notice that Equations 20 and 21 have only two constraint forces each in them, as opposed to the three forces in Equation 19. The equation for the end joints in a chain will have only two constraint forces because the extremal bodies don't have a joint on their "far" side (or they wouldn't be very extremal, now would they?).

Now for a bit of matrix magic. Equation 20 contains f_{c1} and f_{c2}, Equation 19 contains f_{c1}, f_{c2}, and f_{c3}, and Equation 21 contains f_{c2} and f_{c3}. Each of these equations depends on one or more of the other ones. This is the mathematical expression of our statement above that the constraints must be solved simultaneously. We can construct a single large matrix equation that contains all of these equations by stacking them up, like so:

$$\begin{bmatrix} \mathbf{A}_{1u1} + \mathbf{A}_{1d1} & -\mathbf{A}_{1d2} & 0 \\ \mathbf{A}_{2u1} & \mathbf{A}_{2u2} + \mathbf{A}_{2d2} & -\mathbf{A}_{2d3} \\ 0 & -\mathbf{A}_{3u2} & \mathbf{A}_{3u3} + \mathbf{A}_{3d3} \end{bmatrix} \begin{vmatrix} f_{c1} \\ f_{c2} \\ f_{c3} \end{vmatrix} = \begin{vmatrix} -b_{1u} + b_{1d} \\ -b_{2u} + b_{2d} \\ -b_{3u} + b_{3d} \end{vmatrix} \qquad \text{Eq. 22}$$

If you perform the matrix multiply in Equation 22, you can see you get the exact equations listed above. Furthermore, Equation 22 is just another $\mathbf{A}f_c = b$ linear system, where $\mathbf{A}$ now stands for the compound matrix in Equation 22, and f_c and b (with no numbered subscripts) stand for the stacked vectors. Instead of a 3×3 system, we now have a 9×9 system, but it's still a linear system and the same rules apply to solving it. Throw Equation 22 into a linear solver, apply the individual f_c vectors back to their appropriate objects, and you've got a constrained system.

From here, it should be pretty clear how to extend this math to an arbitrary number of bodies and constraints. The $\mathbf{A}$ matrix and the associated vectors keep growing, but the structure is exactly the same.

The Linear System Revisited

I said you can solve the $\mathbf{A}f_c = b$ system using a generic linear solver, which is true. However, there are more efficient ways of solving the particular matrix generated by our algorithm that take advantage of its special properties. Efficiency is incredibly important when doing constrained dynamics because linear systems such as $\mathbf{A}f_c = b$ have $O(n^3)$ complexity in the general case, where n is the number of rows in the matrix. This means that every constraint equation you add as an additional row makes the system much slower to solve. $O(n^3)$ complexity is not the kind of slowness that waiting for next year's CPU can fix.

The most important special characteristic of our $\mathbf{A}$ matrix is its sparsity structure. You can see this structure developing in Equation 22, and as you add more bodies and constraints you can see it even better: the constraint submatrices stay on the diagonal of the matrix and its neighboring columns, and the rest of the matrix is zero. This makes intuitive sense given that a constraint depends only on itself (which corresponds to the diagonal element) and its two constraint neighbors (the off-diagonal elements). The official name for the sparsity structure of the $\mathbf{A}$ matrix is "block tridiagonal," for somewhat obvious reasons. What's more, the $\mathbf{A}$ matrix is symmetric, although this fact is not completely clear from our derivation. And finally, it's "positive definite," assuming the constraint equations are well formed. A positive definite matrix is roughly analogous to restricting a real number to be greater than zero, rather than allowing it to be zero or negative. You can learn more about these characteristics in a good numerical linear algebra book.

Taken together, these properties mean we can write (or download) a custom linear solver that will solve our systems in $O(n)$ time. $O(n)$ is definitely the kind of problem that AMD and Intel will make faster every year.

Numerical Accuracy

It's really a shame that all of this math I've presented to you doesn't actually work when you type it into the computer. Well, that's a bit extreme, but the world of floating-point numerics is far removed from that of symbolic equations, and we have to do a bit more work to get them to match up.

IMMERSE YOURSELF IN REALTIME 3D

- Give your best characters the realtime environments they deserve
- Take advantage of MultiGen Creator's sophisticated realtime 3D modeling toolset
- Deliver powerful and optimized 3D content to many platforms
- Produce models and environments with an artist's eye and a programmer's structure
- Use the application realtime 3D professionals prefer
- Author ready-to-run virtual worlds for the most demanding entertainment, urban simulation, and visual simulation applications

For more information contact MultiGen-Paradigm at:

Phone: 408 261 4100
Fax: 408 261 4103
email: sales@multigen.com
http://www.multigen-paradigm.com

Ad reference: CreatorGDMag00

Constructed in MultiGen Creator / Imagery Provided by Dark Suns Studios, LLC.

MultiGen · Paradigm
THE REALTIME 3D COMPANY

Have you checked out the new Gamasutra.com lately?

Gamasutra.com

Find out why Gamasutra.com is a powerful resource for game developers.

More original game development articles. Gamasutra.com showcases technical articles for information about current trends, essays about game design philosophy, or advice for career management. We have hundreds of helpful articles by game developers for game developers right at your fingertips—original content and reprinted editorial from Game Developer Magazine. Gamasutra.com features articles from developers working at some of the leading game development studios—from Activision to Zombie.

Daily industry news.

Keeping up with our fast paced industry is easy with Gamasutra.com. Track personnel changes, company acquisitions and earnings, product and technology announcements, the best-selling games, and more! It's all available on Gamasutra.com.

Access company & product databases, industry job postings, and discussion forums.

As a Gamasutra.com member, you can search our vast company and product databases to help you accomplish your job. Daily updated job postings will help you find your next interesting project. Our discussion forums enable you to exchange ideas and opinions with developers anywhere in the world.

Unparalleled Game Developer's Conference 2000 coverage.

Before, during, and after the GDC 2000, Gamasutra.com covers the event from the game developer's perspective like no other site. Gamasutra.com provides daily coverage of the show including information about new products, show happenings, and the comprehensive postmortems tailored to the needs of programmers, artists, sound designers, and other fields. Plus, you can access the notes from the GDC 2000 speakers.

60K game developers and growing.

Our registered global membership for Gamasutra.com just surpassed the 60,000 mark! We now have members from countries as diverse as India, Ukraine, Poland, Brazil, Australia, and South Africa.

Free access and Membership.

While Gamasutra.com continues to bring innovation and evolution to the gaming industry, one thing remains the same—no fees, for anything. If you would like to be a member of Gamasutra.com, please join at the GDC show booth.

CMP Game Media Group

The major problem is that we're using numerically computed forces to affect accelerations, which are then numerically integrated to find new positions. This algorithm has two big numerical holes in it. For starters, the forces we compute are not going to be exact because of floating-point errors accumulated during the formulation of the $\mathbf{A}f_c = b$ system and its solution. This means the f_c terms aren't going to exactly enforce the constraint equations when they're applied in floating point. To compound matters, the integrator is going to introduce even more numerical errors, since we're using forces to keep a position constraint together. These two sources of error mean the objects will slowly drift apart. At first the errors will be small. If you subtract the positions of the two endpoints of a constraint, you'll see the result is not exactly the zero vector after a few steps. Eventually, the objects will have drifted far enough apart so you can see the gap. This is bad.

There are many ways of dealing with this drift, and we don't have the space to talk about any of them in depth. I favor a method called Baumgarte Stabilization, which basically places tiny springs on the joints that are adjusted to suck up the numerical error as it develops. The springs don't actually provide any physical support (the f_c terms still do that), but they do a great job of keeping the joints together in the face of floating-point errors. The sample application implements Baumgarte Stabilization to fight the drift problem. It's easy to implement and it works well.

Other methods include directly correcting for the drift in position space, and other techniques. Now that you know the math behind constrained dynamics, you'll have no trouble following the numerical accuracy discussions in the books referenced on my web site.

Miscellanea

Phew! That's a lot of equations, but we've accomplished a lot. We've actually accomplished even more than we set out to, because all of this math is valid for completely general constraints. When I say general, I mean it in two ways: topologically and in terms of the joint types.

Topologically, our constraint-centric viewpoint means we can have as many constraints coming off a rigid body as we like. We'll need to modify our notation slightly to support this, and the location of the elements in the Equation 22 matrix will change a bit depending on the interconnection of the bodies, but making an octopus would be no problem, even though the root body has eight constraints on it.

As far as joint types go, Equation 3 is just one of an infinite number of acceleration constraints this math can enforce. Again, pieces of the derivation change, but the overall structure stays the same, regardless of whether you're simulating spherical joints such as Equation 3, or hinges, or prismatics, or whatever. Look at the references on my web site for books about writing different constraint equations, or give it a try yourself. The important thing to remember is to get it clear in your head what you'd like the joint to be, and then write down an equation that describes that joint. Differentiate it then plug and chug.

Hopefully, with the general interest in physics for games that's been growing for the past few years, we'll start to see a lot of special effects using real, consistent physics. Then, when everybody's comfortable with the math and implementation issues, we can start to work on integrating physics with game play and game physics can finally live up to its potential. ■

FOR FURTHER INFO

My dynamics web site, including the ponytail sample app, articles, references, and more:

http://www.d6.com/users/checker/dynamics.htm

Acknowledgements

I forgot last time to thank Lisa Washburn of Vector Graphics (http://www.vectorg.com) for the model of the head and the ponytail pieces.

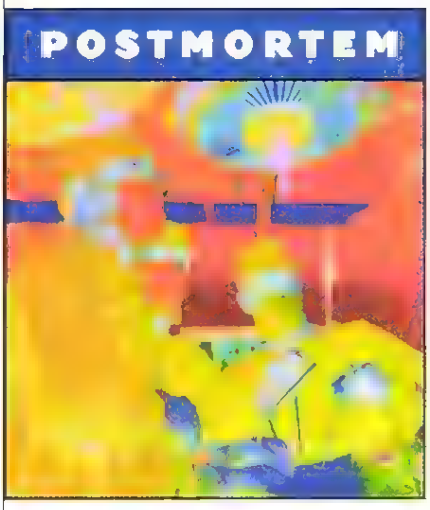
Sports Games Producer

NINTENDO OF AMERICA, the world leader in video game entertainment, invites highly-talented and experienced sports games producers to consider this exciting opportunity at our Headquarters in REDMOND, WASHINGTON.

This key individual will manage and oversee software development; help approve new concepts submitted by developers; and work with second party developers on concept implementation. REQUIREMENTS: Minimum of 5 years sports experience (preferably football, basketball or baseball) at the Associate Producer level or higher; proven game knowledge spanning all consumer and arcade hardware platforms over the last decade (console sports experience such as Dreamcast or PlayStation a plus); excellent project management, supervisory, and written/verbal/interpersonal communication skills; and the ability to travel both domestically and internationally.

Nintendo offers a compensation package that includes excellent benefits. Please forward resume with salary requirements AND cover letter specifying titles worked on and in what capacity, to: **Nintendo of America, Human Resources Dept., Attn: CM/SP, P.O. Box 97033, Redmond, WA 98073. E-mail: recruiter@noa.nintendo.com. FAX: 425/882-3585. Jobline: (425) 861-2170.** For more information or to APPLY ONLINE, visit: **www.nintendo.com.** Nintendo is an equal opportunity employer supporting a drug-free workplace.

Nintendo®



Turbine's Asheron's Call

by Toby Ragaini

ASHERON'S CALL is a statistical anomaly. In an industry where cancelled games and dashed hopes are the norm, this project seemed one day away from certain failure for nearly its entire history. And yet, thanks to the visionary foresight of a handful of people, a healthy dose of luck, and incredible conviction from both the development team and publisher, it made it to store shelves and has received a great deal of critical acclaim.

In May 1995, I walked into a small suburban home in southern Massachusetts and met my new co-workers. Having left my previous job at a genetics lab, I expected nothing more than an interesting summer project as "A Game Writer." Little did I realize what was in store for me and this start-up company called Turbine.

Toby Ragaini leads Turbine Entertainment Software's design team and was the lead designer of ASHERON'S CALL. He is currently working on Turbine's next-generation massively-multiplayer game.

Having filled every nook of a residential home with PCs, an enterprising group of about ten developers was already busy working on the game that would one day become ASHERON'S CALL. Although not a single one of them had professional game development experience, I was immediately impressed with their enthusiasm and dedication. After introductions, I was told to scrounge around for a desk. Upon securing an end table and a plastic lawn chair, I sat down and started meeting with various team members to figure out just what this game was all about.

What was described to me was something that nearly every computer game geek is by now familiar with: a 3D graphical MUD. A persistent fantasy environment where hundreds of players could explore the land, defeat monsters, form adventuring parties, delve into dungeons, and complete quests. I'm not sure why anyone thought it was possible. We had no office, no technology to speak of, and no publisher. And I was being paid $800 a month. Yet from these humble beginnings, something truly wonderful was created.

Dynamic load balancing on the server gave ASHERON'S CALL an expansive, seamless game world that required no load times.

The development team was divided into functional departments. Tim Brennen, a Brown University dropout who had helped develop Windows NT as a Microsoft intern, led the engineering team and would go on to design the server, networking, and character database. Chris Dyl, a former physicist turned programmer, would develop the 3D graphics engine and server-side physics. Andy Reiff, also a Brown alumnus, would later round out the engineering leads as the game systems programmer, responsible for implementing all of the game rules systems and functional interactions in the game world. All of the game's code would be developed from scratch. At the time, this was a fairly easy decision, since licensable game code was pretty much nonexistent in 1995.

On the art team, Jason Booth, a music student with experience using Lightwave, would take on the title of lead technical artist. In this role, Jason bridged the gap between the art and graphics teams, ensuring that the art asset pipeline ran smoothly. Sean Huxter brought his substantial animation and modeling experience to the team as the lead artist.

My own contributions to the team were in the area of game design. As the project grew in scope, my role changed to become that of lead designer. Soon realizing the amount of work required to design a game with the scope of ASHERON'S CALL, I put together a team of designers that envisioned and documented the characters, monsters, history, and timeline of a fantasy world called Dereth. In addition, the design team spec'd all of the game rules and systems necessary to RPGs.

Although the team had no professional game development experience, one invaluable thing that the team *did* have was experience playing MUDs and similar text-based Internet games. Although these games were comparatively simple, the game-play dynamics created in a massively-multiplayer environment are extremely different from a single-player game. MUDs proved to be a very useful model for multiplayer gaming patterns.

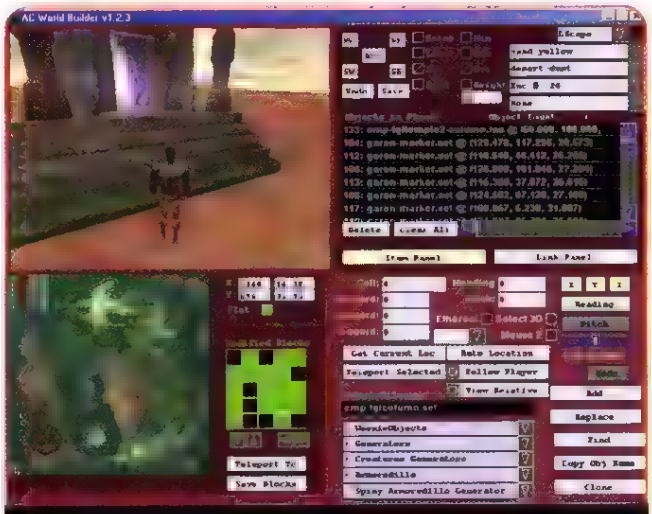
ASHERON'S CALL was initially designed to support just 200 simultaneous players, each paying an hourly fee. Turbine would host the servers, which were originally going to be PCs running Linux. Although in today's market, this sounds ludicrous, in 1995 this was in fact the standard premium online game model. Games using similar models, like Genie's CYBERSTRIKE and America Online's NEVERWINTER NIGHTS, were quite successful at the time. Based on this goal, the original schedule had ASHERON'S CALL shipping in the fourth quarter of 1997.

What Went Right

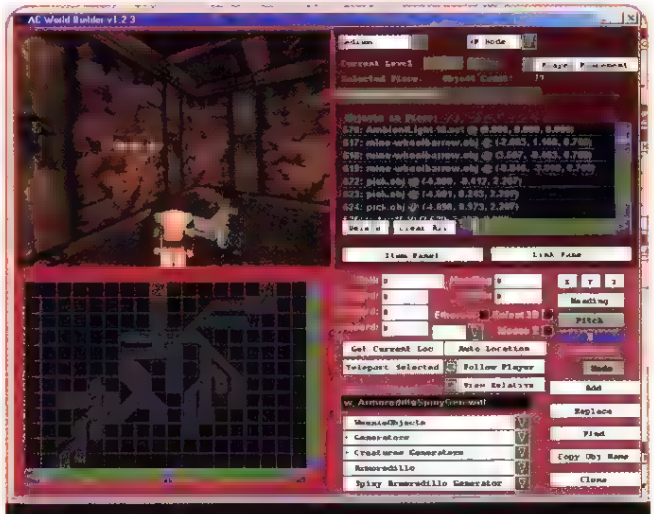
1. STAYING TRUE TO OUR ORIGINAL VISION OF THE GAME. ASHERON'S CALL was a ridiculously ambitious project for an unproven team. Yet despite this naïveté (or more likely because of it), the final product is frighteningly close to the original goal of the project. Of course during that time,

ASHERON'S CALL

Turbine Entertainment Software
Westwood, Mass.
(781) 407-4000
http://www.turbinegames.com
Release date: November 1999
Intended platform: Windows 95/98
Project budget: multimillion-dollar development budget
Project length: 40 months plus 8 months of beta
Project size: approximately 2 million lines of code.
Team size: 30+ full-time developers, including 6 artists, 4 game designers, 15 software engineers, and 5 QA testers.
Critical development hardware: Intel Pentium PCs
Critical development software: Microsoft Visual C++ 5.0, Visual SourceSafe 5.0, Lightwave 5.5, Photoshop 4.0, RAID.



Shot of World Builder decorating an Empyrean ruin on the landscape in the desert. The pillars on that building are hand-placed, as was the portal and the lights that stand outside the door.



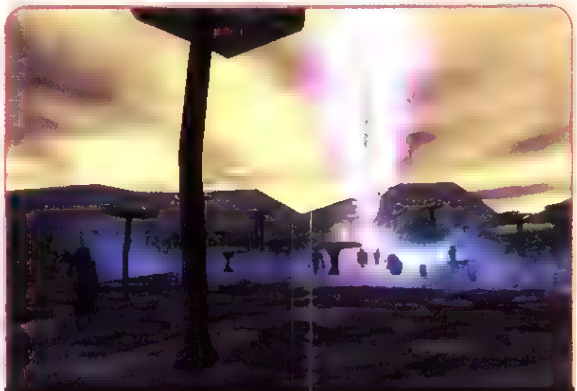
Shot of the in-house dungeon creation tool. Once the dungeon is laid out the decorating can begin. The panel to the right is a list of hand-placed objects in the current piece. Note the mine wheelbarrows and picks on the wall.

Turbine learned lessons in feature cutting, scheduling risks, and compromise. But despite all the missed deadlines, all-nighters, and other disappointments, we are able look back on our shared vision and take pride in that we achieved what we set out to do.

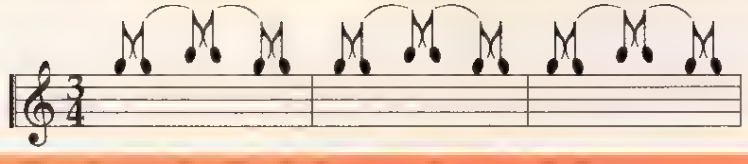
Typically, there exists a master document that describes the overall game concept and goal. Although the documentation at the inception of the game was in fact very sparse, what little that did exist described the fundamental architecture of the game, including its client/server model, dynamic load balancing capabilities (described later), and 3D graphics. In addition, gameplay details such as the allegiance system, magic economy, and the emphasis on social game play are in my notes going as far back as 1995. The team internalized these goals, and a form of oral tradition maintained them in meetings.

Although we didn't know it at the time, ASHERON'S CALL would debut as the third massively-multiplayer online RPG amidst two strong competitors, ULTIMA ONLINE and EVERQUEST. We're often asked if we made any dramatic changes in response to the release of these two titles. In all honesty, the answer is no. If anything, these two products proved to us that our initial technical and game design decisions were correct. Clearly, social game play helped drive the success of these games. This made our game's social systems such as allegiance and fellowships all the more important. It was also obvious that immersion was critical. Instability and pauses were the bane of massively-multiplayer games. In theory, the dynamically load-balanced servers would prevent many of these problems.

In an industry that can be driven by holiday deadlines, marketing hype, and cutting corners, it's refreshing to know that ambitious goals can still be rewarded. But it's more than that. While we certainly could have created a less ambitious game, I believe it would have been a detriment to Turbine's competitiveness as an independent development studio. ASHERON'S CALL might have shipped earlier had it been a LAN game or a series of connected arenas, but we would not have the innovative technology and game design experience that today puts Turbine in such a desirable competitive position in the industry. In this way, our team's unwavering vision was handsomely rewarded.



Shot of the landscape test program. Designers ran around the landscape off-line to see what had been added and how it looked. This is a shot of the obsidian plane with mushrooms and crystals, not to mention some creatures lurking near a mysterious power source.

2. SECURING A PUBLISHING AGREEMENT WITH MICROSOFT. In mid-1996, representatives from the newly-formed MSN Gaming Zone were booed by the audience of the first Mpath Developer Conference. Their crime was the prediction that hourly fees were dead and that flat monthly rates would become standard. Our business plan at the time counted on an hourly model, but we recognized the truth to the Zone team's statement. At that year's E3, we relentlessly pursued Jon Grande, product planner on the Zone, in order to pitch him our game proposal and show him our technology demo. At that time, the demo consisted of two PCs connected to each other. One was running the client software, complete with 3D graphics. The other was the server executable. The Zone team was very impressed, and scheduled a visit to our office (we'd since

Whatever the game, LicenseMusic.com has the perfect music.

music tracks. Be it Sunday afternoon or four in the morning, LicenseMusic.com's search and licensing engine makes it simple and convenient to find music for that alien attack with. No salespeople to haggle with. Next time, put your game to music and mind to rest by visiting or by calling

LICENSE MUSIC.COM

Just what you want to hear.

©2000 Online Music Company, Inc. LicenseMusic.com is a service mark of Online Music Company, Inc. All rights reserved.

Different types of combat were developed independently, causing complications.

moved into an actual office space south of Boston). Soon after the visit, Microsoft agreed to enter into a publishing agreement with Turbine, secured initially with a letter of intent. The actual contract arrived six months later, but the letter of intent granted us an initial milestone payment and enough certainty to schedule the milestone deliverables. This was the start of a long, sometimes tumultuous, but ultimately fruitful alliance.

After we secured the contract, the division of labor was discussed. As the developer, Turbine was to design the game, engineer and implement all of the code, generate all art assets, create a QA plan, and perform testing on all game content. With its pre-existing Zone platform, Microsoft was responsible for code testing, billing, and ongoing server operations. Fundamentally, this meant that while Turbine would create the game, the day-to-day operations of the ASHERON'S CALL service would be entrusted to Microsoft.

One thing that Turbine successfully negotiated for was the rights to our source code. Besides the team, we knew that our massively-multiplayer technology was going to be our single most valuable asset. In addition, we agreed to a one-title deal that gave us the flexibility to pursue other development deals as opportunities arose. In this way, we ensured that Turbine would remain independent and effectively in control of our own destiny.

In many respects, Microsoft proved to be an ideal partner for Turbine. Like Turbine, the Zone was a start-up organization, and was eager to prove itself. The Zone was pioneering a new type of business, with a business model new to Microsoft, and this placed the managers of the Zone in a position where they could afford to take risks. And while ASHERON'S CALL ultimately validated Microsoft's belief in Turbine, at that point Turbine was certainly a risk.

Besides the obvious funding issue, Turbine benefitted from its partnership with Microsoft in other ways. We had free access to Microsoft development tools like Visual C++, Visual SourceSafe, and a bug-tracking database called RAID. We learned a lot about professional software development from Microsoft as well, such how to create an efficient build process, manage code source trees, and organize effective test cycles on the daily builds. Finally, we gained prestige by working with one of the most respected software companies in the world. Having Microsoft as a partner gave us a lot of credibility and put us in a much better position to pursue funding and make critical hires, two incredibly important objectives for a small start-up company.

3. REUSABLE ENGINE AND TOOLS. Massively-multiplayer games require a fundamentally different architecture from that of single-player games, or even multiplayer LAN games. Beyond the graphics engine, user interface, and other elements of a typical game, persistent massively-multiplayer games generally require a centralized server, networking layer, user authentication, game administration tools, and a host of other technologies.

Early on, Turbine recognized that many of these technologies would be required by any massively multiplayer game, and could perhaps be generalized enough that they could be reused in different massively-multiplayer titles. At the time, this was an unusual premise for a game developer; typically, source code was thrown out at the end of a project, and the idea of licensing a 3D engine like Quake was still a long way off. From our perspective it just made good business sense to leverage our R&D as much as possible. Since so much of our development budget was devoted to creating these key technologies, we made every effort to keep the technology modular and data-independent.

This modular architecture has since proven to be a tremendous win for Turbine. We've been able to prototype new game concepts rapidly by changing data while keeping the server executable nearly unchanged. Not only has this helped us get new business, it has also proven to be extremely useful for in-house play testing and constructing proof-of-concept demos.

Currently we are investigating the potential of licensing our technology. While we continue to advance the code base, we have placed some emphasis on productizing the Turbine engine. From a business perspective, this is a very desirable source of revenue. We can leverage our R&D efforts and development costs, while advancing the engine that our own future products will use.

In addition to the ability to reuse code, Turbine's modular emphasis extended to the way content is created for the game world. As development on ASHERON'S CALL progressed, we quickly came to realize that populating a game world the size of Dereth was going to be a monumental task. By this time, we knew our competitors were hiring

NOW PLAYING

Thanks to Pandemic Studios and technology from Intel and Digimation.

Pandemic Studios used MultiRes technology in both Dark Reign 2 and Battlezone II to offer high frame rates without sacrificing either game's incredible graphic detail. Now you can use the same technology in your own productions. Digimation has combined their experience in 3D software with Intel's scalable graphics technology to create five indispensable tools for game developers, 3D artists and web site creators. Each tool kit contains source code and library files ready to enhance your project.

MULTIRES 2 MultiRes gives game developers and artists the ability to create scalable 3D models that automatically adjust resolution according to distance, target frame rates and available processing power.

SUBDIV RT SubDiv RT provides users with a method of increasing the polygon count of low-resolution 3D models by adding polygons in real-time. This effectively means a user can create, and initially deliver, a low polygon count model that is compact in size and then dynamically increase the resolution when it is displayed, providing a richer visual experience than the smaller sized model.

ANIMATE RT Combine real-time skeletal deformation and IK and you get Animate RT. An object's motion can be pre-authored with other software packages such as Character Studio® or Bones Pro 2 inside of 3D Studio MAX®, or generated in real time by the Inverse Kinematic (IK) technology that is included in Animate RT.

PARTICLE RT Particle RT gives content creators a complete particle system for generating thousands of particles in real-time. These particles can then be affected by physics and can have bitmaps added to achieve widely different effects from billowing smoke to fiery explosions.

TOON RT This exciting real-time render technology allows users to transform and render their 3D models in a variety of non-photorealistic methods such as cartoon, pen & ink and sketch!

DIGIMATION™
The Digital Animation Company

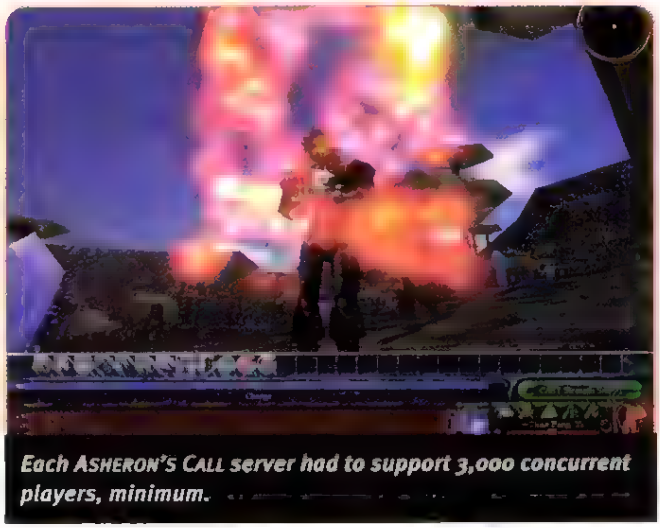
www.digimation.com
1.800.854.4496 or 504.468.7898

WITH intel Optimizers
SCALABLE 3D GRAPHICS

SEE INTEL AND DIGIMATION AT BOOTH #1118 AND #1436

teams to design individual levels and create content manually. This seemed less than optimal to us, and furthermore we didn't have the resources to hire a large content team.

Instead, we created a series of world-building tools to maximize our efforts. The first kind of tool allowed artists to create vast chunks of game environment (represented as a grayscale height map) with each stroke of their brush. Random monster encounters and terrain features such as trees and butterflies could also be placed using this method.



Each ASHERON'S CALL server had to support 3,000 concurrent players, minimum.

We also developed a tool called Dungeon Maker to create subterranean environments such as dungeons and catacombs. Early on, Jason Booth got sick of hand-modeling the complex level designs he was getting from the design team, so he and user-interface programmer Mike Ferrier created a level-building tool that used an intuitive drag-and-drop interface. This allowed nontechnical designers the ability to create and instantiate dungeons quickly without taking up the art team's valuable time.

An offshoot of Dungeon Maker, World Builder, became a much more advanced tool by the time ASHERON'S CALL shipped. Using World Builder, a content designer could wander around the game world placing houses, decorations, and monster encounters, and even raise and lower the terrain. This proved to be an incredible time-saver, and the amount of landscape content we were able to generate easily quadrupled.

This kind of tool modularity allowed us the ability to update the game world easily with new content, such as new monsters, quests, items, and adventure locations. Thanks to monthly content additions, ASHERON'S CALL "events" can propel an overarching story forward and involves players in all areas of the games. So far these events have proved to be a huge success. Players feel like they are part of a living, breathing world, and are more likely to stay involved in the game for longer periods of time.

4. PAINLESS LAUNCH. When the first few thousand players began pouring onto the production servers, we were certain that there would be all sorts of catastrophes. We had watched our competitors suffer similar calamities, and we had resigned ourselves to accept this rite of passage. To our surprise, nothing went wrong the first day. We were delighted by just how stable and uneventful the retail launch was. Everything went without a hitch.

This stability was due to effective beta testing, intelligent project management, and insightful data-center equipment deployment. Here's how it worked. During beta, both Microsoft and Turbine testers submitted bugs into RAID. In addition, user-submitted bugs were tracked by the Microsoft team and were added into RAID if they were deemed important. Server performance metrics were one of the key goals towards meeting our shipping requirements. Each server had to maintain a minimum level of performance, given a concurrent user base of 3,000 players. To meet this metric, a few changes were in order. The server-side physics was modified to use a more simplified collision model. In addition, a faster "clean-up" cycle for objects dropped on the landscape was implemented. Having made these changes, we were able to meet the aggressive server metrics and our server software has since proved to be nearly bulletproof. In fact, for the first several weeks, the server software did not crash once, which was a major accomplishment considering the tech-

Audio Software Engineer – Product Support

Combine your love of video games with your desire for programming challenges! NINTENDO OF AMERICA, the world leader in video game entertainment, offers this exciting opportunity at our Headquarters in REDMOND, WASHINGTON.

Requirements: BS in Computer Science or equivalent; 5+ years of solid programming experience in C, C++ under windows or embedded systems (consoles); knowledge of audio/video compression techniques (ADPCM, MP3, MPEG2), general MIDI and MIDI sequencers (Cubase, Cakewalk), audio topics (Dolby Surround, Dolby Digital, DLS, MPEG4, Direct Sound) and DSP (algorithms and programming); and excellent written, verbal and interpersonal skills. Musical talent is a plus!

Nintendo offers a compensation package that includes excellent benefits and the opportunity to work in a fun, fast-paced team environment. Please forward cover letter and resume with salary requirements to: **Nintendo of America, Human Resources Dept., Attn: CM/ASE, P.O. Box 97033, Redmond, WA 98073. E-mail: recruiter@noa.nintendo.com. FAX: (425) 882-3585.** For more information or to APPLY ONLINE, visit our website at: **www.nintendo.com. Jobline: (425) 861-2170.** Nintendo is an equal opportunity employer supporting a drug-free workplace.

Nintendo®

"For over ten years we have been using LightWave for the development of a countless number of games. We use LightWave because it is the best combination of power, convenience and price for the job."

Matt McDonald
President/CEO of Vision Scape Imaging, Inc./The Lab

LightWave® [6]

LightWave builds character

"Come see us at Booth #1714 at the Game Developer Conference"

Bring life to characters better and faster with the new animation tools in LightWave [6]. Endomorphs allow you to create a single model with embedded morph states, which provide the ability to add or remove detail from your base model without effecting the embedded targets. Your model now contains nonlinear data for complex facial movements and organic animation.

Skelegons are a new type of polygon that look and act like traditional 3D bones. Skelegons provide characters with an intact skeleton that is automatically updated as changes are made to the model. Skelegons can easily be transplanted from one character to another or you can use any of LightWave's modeling tools to create a completely new skeleton with Skelegons.

Also new to LightWave [6] is the world's fastest and most accurate hybrid Inverse/Forward Kinematics engine. Combining the IK system with Endomorphs, Skelegons, the powerful new curve editor, and low-poly with subdivision surface modeling, LightWave [6] is the ultimate tool for single mesh character animation.

Designed with the game programmer in mind, LightWave [6] provides open, expandable object and scene data formats. LightWave [6] also offers precise vertex level model control, flexible and exact UV editing, an L-Script scripting system and a new plug-in SDK.

The joy of [6], all the power, all the control, open, fast, fun

For more information on LightWave [6]
Call 1-800-862-7837 or visit our website at
www.newtek.com

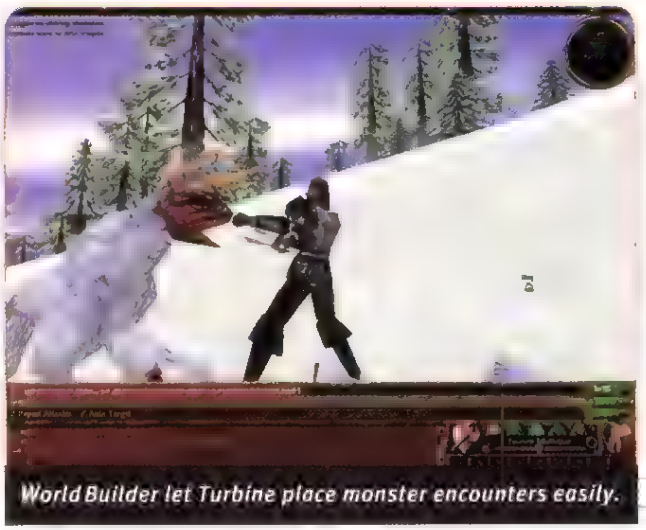
To reach NewTek from outside the U.S. call (210)370-8000

LightWave 3D® is a registered trademark of NewTek, Inc. Endomorphs and Skelegons are trademarks of NewTek, Inc. © NewTek, 1999. All rights reserved.

NewTek

nical problems evident in other massively-multiplayer games.

Our retail launch was a staggered affair. Initially, only two "enthusiast-oriented" retail chains received shipments of ASHERON'S CALL boxes. This allowed our die-hard fans from the beta testing program to get copies, but prevented the deluge that would have occurred had we been in the larger, more mainstream retail stores. While it would have been exciting to see massive sales on day one, I believe that this gradual approach was a smart move.



World Builder let Turbine place monster encounters easily.

5. SEAMLESS ENVIRONMENT USING DYNAMICALLY LOAD-BALANCING SERVERS. One the most impressive features of the Turbine engine is the continuous outdoor environment. This is made possible thanks to dynamic load balancing, which is a scalable server-side architecture. The easiest way to appreciate the need for dynamic load balancing is to consider the following scenario.

Imagine a hypothetical game world that is divided into four servers, each of which corresponds to a geographic area in the game world. With a static server architecture, if everyone in the game world decides to go to the same area, that one server's performance would be dramatically impaired, while the three remaining servers would effectively be idle, completely unaware of their overtaxed brother.

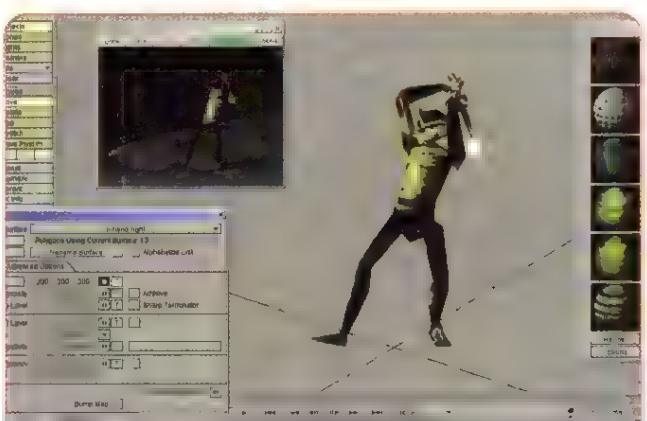
Dynamic load balancing solves this overloaded server problem. Instead of assigning a static geographic area to each server, the individual servers can divide up the game world based on the relative processor load of each server. In the previous example, instead of remaining idle, all four servers would divide the load equally among themselves, ensuring the most efficient use of the hardware's processing capacity.

Dynamic load balancing allows a very free-form environment where players can travel wherever they want with very few hard-coded limits. But in order for the graphics engine to accommodate the seamless nature of the server, we couldn't allow a "level loading" pause typical in many 3D games to interrupt the game play. To avoid level-loading, the geometry team headed by Chris Dyl engineered a unique rendering engine that constantly loads data in the background, and draws objects at far enough distances so as to minimize obvious "popping" effects and without having to rely on a fogging effect to hide the clipping plane.

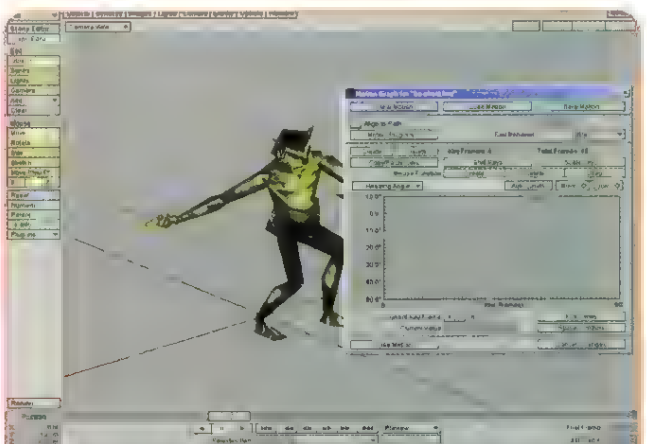
What Went Wrong

1. POOR SCHEDULING AND COMMUNICATION. For most of its early history, ASHERON'S CALL was the victim of poor project management. During the last year of development, a management reorganization took place that salvaged the project. Depending on how far back you look at the schedules, ASHERON'S CALL was either one to two years late. This is attributable to a number of reasons, some of which I will explain momentarily.

When Microsoft and Turbine entered into the development agreement, neither side had any idea of the scope of the project. An initial list of milestones was drawn up by the Microsoft product manager and our development leads. Unfortunately, after the second milestone, deadlines were consistently missed. A lot of this was due simply to underestimating the time required for development tasks. This created a domino effect as we continually played catch-up, trying desperately to make up for lost time.



This is a shot of the Banderling being animated. He is performing a large swing to the front, and is caught in mid-backswing. The weapon is rooted out of the scene when the animation is preprocessed, as is the floor. Note the movement path of the weapon as it strikes.



This is a shot of the Banderling being animated. The motion graph shows the keyframes and the spline movement of the Banderling's chest as he makes a swing with a weapon.

With Other Sound Cards You Hear Him Empty His Weapon.
With Monster Sound MX400 You Hear Him Empty His Bladder.
S3's Diamond Monster Sound MX400 will rock your PC audio world. Based on ground-breaking technology from ESS and Sensaura, the Monster Sound MX400 adds a whole new twist to your gaming experience with scorching 3D positional audio, so now you hear sounds on a whole new axis—above and below you. And with true quad output and Dolby Digital* surround sound*, you create the ultimate PC home theater. Plus you can play, download, store and manage the hottest digital audio formats on the Internet or copy tracks from your CD collection to build your own high quality MP3 files! SO UP YOUR AUDIO WITH MONSTER SOUND MX400—truly in a class by itself!
Screenshot of Slave Zero courtesy of Accolade.
*Requires Dolby Digital receiver
ESS
DIAMOND
Monster sound MX400
Up Your Audio!
Powered by ESS Canyon3D PCI Audio Technology
Screaming 3D Positional Audio
True Quad Output
Dolby Digital* 5.1 for Home Theater and Music
S3
DIAMOND MULTIMEDIA
AUDIO GRAPHICS COMMUNICATION HOME NETWORKING
www.s3.com

We'd like you to take on 30 million Americans.

Believe it or not, that's how many play dice every day. Which is why we've created a set of real dice that work seamlessly with both computer and video games.

CIBRO Technologies Ltd.

Are you ready to roll?

www.cibrodice.com
email: info@cibrodice.com

This schedule free-fall continued into 1997 and forced us to re-evaluate the feature set. Unfortunately, feature cuts were made without considering the impact on the playability of the game. Ultimately, most of these features were added back into the game anyway, which took additional time due to the reallocation of team resources. The lesson here concerns the value of effective scheduling. Identify the risky areas in your schedule early, figure out the dependencies, and make sure you pad the time estimates for tasks.

Communication between Microsoft and Turbine was also a major factor. The teams were separated by about 3,000 miles and three time zones. Although weekly conference calls were scheduled, they lacked the collaborative mentality necessary for maintaining a successful relationship. E-mail threads were either ignored or else escalated into tense phone calls, and in some cases the bug-tracking database (RAID) was not used effectively.

Clearly, everyone would have benefited from more face-to-face time. E-mail — and even conference calls — are poor media for managing new and sensitive corporate relationships, especially ones between companies with such different corporate cultures. From a developer's perspective, it's always easy to blame the publisher for unrealistic expectations and bureaucracy. What's important to realize is that it is everyone's obligation to communicate expectations and problems before they escalate to the point of being a crisis.

2. INEXPERIENCED DEVELOPMENT TEAM. None of the senior developers at Turbine (including me) had ever shipped a retail PC game. *None*. Many of the employees were students immediately out of college, or even college students completing a work-study program. This obviously was the source of several severe problems in the development of ASHERON'S CALL.

It was nearly impossible for team leads to give realistic schedule estimates for tasks, since few of us had experience in professional software development. It was also initially difficult to get different teams from the programming, art, and design departments to communicate regularly with each other. The collegiate atmosphere made it very difficult for decisions to be made; meetings would happen and resolutions would seemingly be agreed upon, only to have those same questions asked in a subsequent meeting. No one likes unnecessary bureaucracy and giving up creative freedom, but ultimately one person needs to be given the authority to make a decision and hold people to it. A good supervisor takes into account the opinions of everyone involved; design by committee simply does not work.

Obviously, having a seasoned and experienced development team has innumerable advantages. While it's not critical that everyone on the development team have professional experi-

Early early shot of a mock-up UI that had very limited functionality. Some of the buttons worked, but mainly this was a prototype pasted over the 3D window, which was working at the time. The upper-left set of buttons were macros. The paper doll was hand drawn, and did not reflect the look of the character.

This is an incredibly early shot of our Character Generator screen. Much of the interface has remained unchanged, but the same cannot be said of the faces. This was probably version one of the face images. Turbine has gone through a couple of versions since then.

Breakthroughs. Interactions. Insights. Visions. Applications. Adventures.

2000 SIGGRAPH

The world's premiere conference and exhibition on computer graphics and interactive techniques.

Conference 23-28 July 2000
Exhibition 25-27 July 2000
www.siggraph.org/s2000/

Images (top to bottom): Civilization of Fruit: Martyred Apple © 1998 Patricia Swain; Still Life © 1998 Scott Curtin, Ringling School of Art and Design; Rollie Pollie Ollie © 1999 Nalvana; The Space Field Series: Comet Nebula © 1997 Victor Raphael

Sponsored by ACM SIGGRAPH

ence, at the very least team leads should have some form of professional experience. As it was, Turbine had to get by with raw talent, unabashed enthusiasm, and simply not knowing any better.

3. NO FEATURE ITERATION DURING DEVELOPMENT. Many weaknesses of ASHERON'S CALL at launch stemmed from the methodology we followed for feature completion. Features were scheduled by milestone and were expected to be completed in their entirety before other features were worked on. While this approach may work for more typical software applications, PC games rely on a host of interrelating systems that cannot be implemented in a vacuum.

An example of this involved our melee combat system. This game feature was completely spec'd and implemented long before magic spells worked within the game, under the misguided assumption that it saves developer and test resources not to have to revisit completed features. Clearly, these two game systems needed to be tested and balanced in stages alongside each other, not independently.

Another example of this problem occurred during beta testing. A massively-multiplayer game cannot be considered adequately tested until thousands of players have participated in the game world for at least a few months. The first time ASHERON'S CALL was exposed to this many users was when it went into beta testing. Unfortunately, we were placed in a code freeze situation during the beta test, and only the most serious bugs were fixed.

Both Microsoft and Turbine recognized many serious game balancing problems during beta, but at that point it was extremely difficult to make changes. This can be attributed to our tight schedule, but earlier beta tests would have accelerated the bug-finding process and resulted in a better balanced game. On future projects, Turbine is deploying a more iterative implementation process where rapid prototyping and early play-testing is encouraged.

It was difficult to play-balance the game since that aspect of development took place after the code was frozen.

4. AN AMBITIOUS PROJECT LACKING FUNDAMENTAL UNDERLYING TECHNOLOGIES. As one of the first massively-multiplayer 3D games, ASHERON'S CALL was a bold undertaking. Several core components were still theoretical when the project was planned. Things like dynamically load-balanced servers and continuous, uninterrupted outdoor environments were still unproven concepts when we committed to them for ASHERON'S CALL. Furthermore, we had to create our own 3D graphics engine, a latency-friendly network layer, and physics and game rule systems that would all work within a client/server model.

We learned very quickly why there hadn't been a game like ASHERON'S CALL before us: It was damned hard to develop such a game. I don't think committing to a less aggressive feature set was the right solution, though. Instead, we should have acknowledged up front that R&D efforts are fundamentally hard to schedule, and been more flexible with our development schedule. With this in mind, we could have created more realistic estimates and done a better job managing expectations within and outside Turbine.

5. NO DOCUMENTED HIGH-LEVEL FEATURE STATEMENT. Because ASHERON'S CALL had such a long and evolving development cycle, it was difficult to keep all the documentation up-to-date. To compound the matter, the project never had an official feature set as part of the development contract with Microsoft. The technical design document process and high-level feature overviews were basically skipped. This created severe problems when it came to prioritizing which features were important. We constantly had to justify features, and we had no documentation to fall back on to resolve our discussions.

Without a high-level vision statement it was also very difficult to educate new employees about the game. There was a sort of oral tradition to initiate new employees that had been passed down for so long that it just became part of our company's culture. This was partially possible because the concept of a 3D graphical MUD intuitively made sense to a lot of people. Unfortunately, it was very difficult to explain what ASHERON'S CALL was about to people who didn't understand this concept or had their own ideas about how things should be done. Having a documented vision statement and a description of the high-level feature set is absolutely essential for any title.

A Unique Company Résumé

ASHERON'S CALL was a tremendous learning opportunity for Turbine and Microsoft. Despite all the problems and setbacks, ASHERON'S CALL is a success story. The game has been well received by PC game enthusiasts as well as the majority of the game industry press. The fan support for ASHERON'S CALL is overwhelming, and players routinely spend more than six hours a day logged into the game world.

In addition, Turbine is now in a very desirable position, being one of only a handful of developers (and the only independent studio) that has successfully created a massively-multiplayer title. Industry analysts predict that online games will be the fastest growing segment of entertainment software. With its reusable architecture, robust toolset, and (now) experienced developers, Turbine intends to remain at the forefront of massively-multiplayer gaming. ■

Play with more intelligence.

The fun is back. Kids of all ages are into video games like never before. In fact, microchip technology has transformed almost everything about home entertainment, making products better and smarter than ever.

STMicroelectronics supplies billions of microchips for a broad spectrum of digital consumer products, particularly for advanced systems, such as digital cameras, DVD, HDTV, set-top boxes, and PCs.

That makes ST a major player in the digital revolution.

We put more intelligence into everything.

STMicroelectronics, formerly SGS-THOMSON • A world leader in semiconductors for consumer products, automotive, telecommunications, computer peripherals, industrial automation and control systems • Ninety-five locations on five continents • www.st.com

Where Talent Meets Opportunity

Sometimes it's all about what you know.

In an industry where opportunities pass by before most knew they even existed, it's crucial to never lose sight of where you want to be.

Sometimes it's all about who you know.

Whether you hope to build a career with one of the video game industry's premier companies or to create your own household name from scratch, it's essential to connect with people who can help you realize your potential.

Interact is where what you know and who you know come together.

Interact is a multi-million dollar talent agency specializing in the placement of individual game developers and the representation of independent studios and interactive rights for many high profile licensed properties. We offer the most well-connected staffing network in the game business. With over seven years in the industry, we have the inside knowledge and experience to put you within reach of virtually any opportunity you seek. Put our resources to work for you.

831 S. Douglas St. Suite 119
El Segundo, CA 90245
TEL: 310-643-4700
FAX: 310-643-4750
TEL: 800-333-5751
WWW.INTERACTJOBS.COM

INTERACT

"We want people who can break the rules."
Artists, Programmers, Animators, Designers
Positions Available in Silicon Valley, San Diego and Chicago
Check out hot job opportunities at midway.com
See us at the GDC in Booth 1835

LET THE GAMES BEGIN!

NEW from Coriolis Technology Press

Learn the Best Practices for Game Design and Programming with...

GAME ARCHITECTURE AND DESIGN

ISBN: 1-57610-425-7

Price: $49.99 U.S. • $73.99 Canada

Written for PC, Macintosh, and Unix platforms, this book uncovers object-oriented design, core design, gameplay, game balance, and provides real-life case studies.

Discover the Insider Secrets of Today's Professional Game Developers with...

3D GAME PROGRAMMING WITH C++

ISBN: 1-57610-400-1

Price: $49.99 U.S. • $73.99 Canada

Learn to create games for PC platforms and graphics accelerators and harness the power of DirectX. Includes a foreword by Andre LaMothe and a full source code game engine.

Books include CD-ROM packed with valuable tools!

Available at Bookstores and Computer Stores Worldwide

Telephone: 800-410-0192
Fax: 480-483-0193
International: 480-483-0192
In Canada: 800-387-9776

CORIOLIS Technology Press

www.coriolis.com

©1999 The Coriolis Group Prices subject to change. CD100 1099

Programmers Designers World-builders Artists

Make the game! Share the rewards!

We work hard, we play hard, and we develop the best games in the world!

Published
"Duke Nukem: Time To Kill"
"Die Hard Trilogy 2"
Both "Rugrats" PSX titles

Current projects
"Duke Nukem: Planet of the Babes"
"DangerGirl"
plus 2 "Unannounced PlayStation2" titles

Only the best need to apply!

Check out WWW.N-SPACE.COM for more information on these opportunities and our sweet royalties sharing package!

n-Space, Inc.
Attn: Human Resources
7035 Grand National Drive
Orlando, FL 32819
fax: 407.352.5571
email: hr@n-space.com

n SPACE
DEVELOPER OF INTERACTIVE VIDEO GAMES

BUILD THE FUTURE TODAY

Can you say *Massively Multiplayer?*

Interested in *Broadband?*

Want to help shape the future of *interactive entertainment?*

Sony Online Entertainment is looking for creative thinkers, visionary dreamers and committed enthusiasts to work with the hottest game brands online today: From *JEOPARDY! Online* and *Wheel of Fortune Online* to the ground-breaking *EverQuest*!

At *Sony Online Entertainment*, we know how to satisfy people's passions for what they love most—Games! We work with the best in the industry, create the best games and have the best time doing it!

We have immediate openings for talented individuals in **Los Angeles** and **New York**!

- Game Designers
- Software Engineers
- Database Engineers
- Unix System Administrators
- Art Directors
- Graphic Designers
- Producers
- Associate Producers
- Integration Producers
- Product Managers

For a complete listing check the web at **www.station.sony.com**

E-mail resumes to: **NY** - jobs@station.sony.com
LA - resumes@spe.sony.com

Or mail to: Sony Online Entertainment Inc.
Attn: Human Resources
550 Madison Avenue, 7th floor
New York, NY 10022

Sony is an Equal Opportunity Employer.

The Station™
@ sony.com

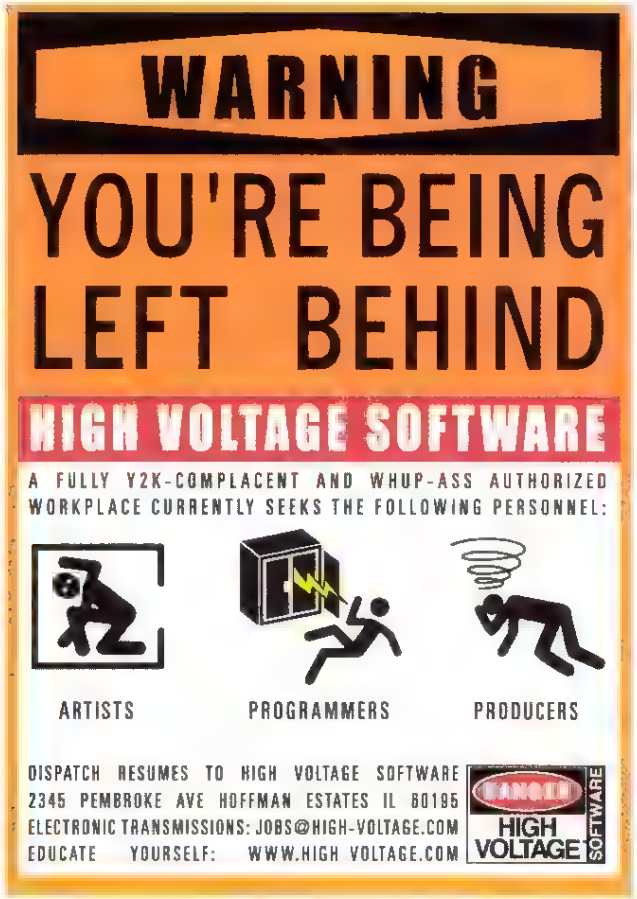
Imagine an IT job
where you get world-class
training and your department
is never under-staffed.

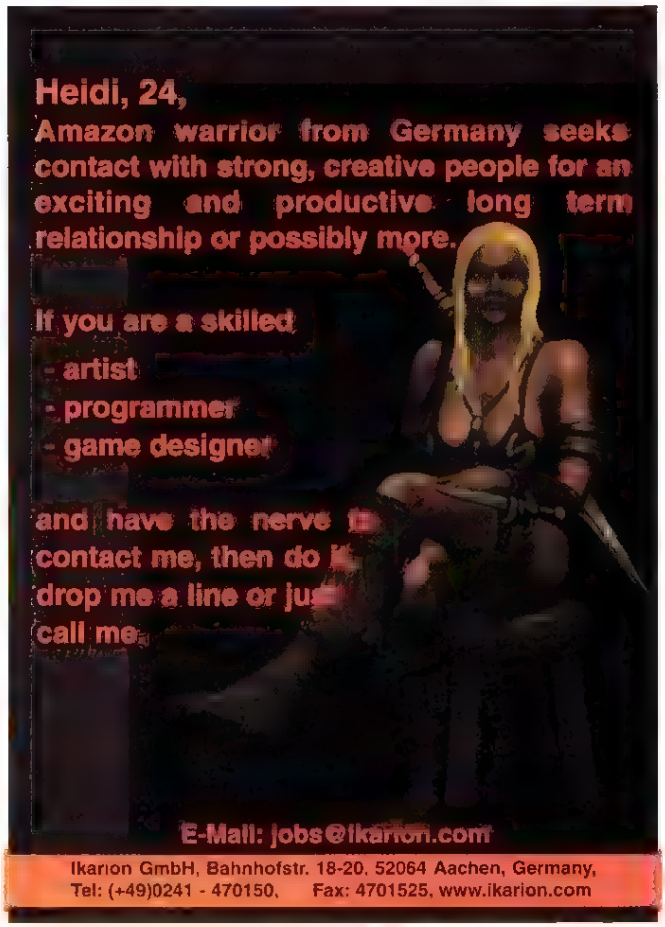
Or go to
dice.com and
actually find one.

dice.com
High tech jobs online

110,000 high tech jobs, including your next one.

Free Sodas.
Bagel Fridays.
Oni and Halo.
(Development doesn't get any better than this.)
Bungie Software is seeking
PROGRAMMERS, MODELERS,
ANIMATORS, DESIGNERS
and TEXTURE ARTISTS.
(Texture Artist Position at Bungie West)
For more info, surf to jobs.bungie.com or visit us
at the GDC Job Fair Booth #1937. To apply, send your
resume along with demo reels or other materials to either
jobs@bungie.com
or
ATTN: JOBS
Bungie Software
350 W. Ontario 7th Floor
Chicago, IL 60610
BUNGIE




Heidi, 24,
Amazon warrior from Germany seeks
contact with strong, creative people for an
exciting and productive long term
relationship or possibly more.
If you are a skilled
- artist
- programmer
- game designer
and have the nerve to
contact me, then do it.
drop me a line or just
call me.
E-Mail: jobs@ikarion.com
Ikarion GmbH, Bahnhofstr. 18-20, 52064 Aachen, Germany,
Tel: (+49)0241 - 470150, Fax: 4701525, www.ikarion.com

www.nxn-software.com/jobs

NxN is a young, international company with offices in London, Munich, Paris and Santa Monica. Leading game developers rely on our unique media asset management system **alienbrain**®

To help as achieve our ambitious expansion plans we are looking for talented people in the following areas:

Sales Manager (Santa Monica, CA)
Pre-Sales Engineer (Santa Monica, CA)
Post-Sales Engineer (Santa Monica, CA)
Associate Marketing Manager (Santa Monica, CA)

NxN Software Inc., Santa Monica, CA
e-mail: jobs@nxn-software.com, phone: (310) 477-0406

SKOTOS
http://www.skotos.net

Berkeley, California

Skotos Tech s an online game compan, committed to creat ng a comm nity of peop e s and world bu ders We w l prov de a massive, mu t p ayer env ronment that a lows p ayers to exp ore many d verse worlds and f they w sh themselves become world b ders w thout re qu r ng knowledge of programm ng

Skotos Tech games are more then text-based adventure games or MUDs our graph ca c ients offer too s to create deep and r ch games w thout os ng the techn ca advantages of a text dom nant med um

Though a start up comp ny Skotos offers exce ent ben ef ts nc ud ng med ca dent a and v s on ns rance 00%-matching 40 k program and stock opt ons

Windows Client Engineer

W ndows eng neer needed to deve op text and graph c c ients for on ne games

Applicant MUST have

- Exten s ve kno w edge of V sual C + & M crosoft Foundat on C asses
- 2 years hands on W ndows GUI programm ng
- Exper ence with Act veX programm ng

Eng neer w des gn web-based Act veX c ent from scratch work w th server eng neer ng team on c ent server protocols and br ng c ent des gn to comp et on Candidates shou d be se f driven and ndependent but able to work with the rest of the des gn team Kn w edge of TCP IP net work programm ng a p us Fam w th MUD adm n and dev a so a p us

Senior Server Engineer

Sen or Server Eng neer needed to develop server for on ne game compan,

Applicant MUST have

- BSCS (or equ valent forma engineering background
- Exten s ve exper ence with an ob ect or ented ang uage
- Sol d understanding of ob ect or ented des gn
- Sol d understand ng of fundamental data structures and algorithms

Eng neer w need to cont nue dev on an exist ng mu t p ayer nteract ve fiction server work ng with a team of other eng neers assoc ated w th a aspects of the project App cant sho d work we with a team and have an interest in the Eng sh language Fam with MUD admin and dev a p us Ex per ence with the LPC or Pike a strong plus

Backend Server Engineer

Backend Server Eng neer needed to ead the development of a user account ng and on ne b ng system Will a so ead deve opment of a porta system for p ayers nc ud ng ema s mp e web pages bud y sts ca endars etc

Applicant MUST have

- C programm ng exper ence
- Database programm ng exper ence
- Exper ence w th shopping carts or other commerce programs
- Exper ence w th comp ex web des gn
- Exper ence programming CGI on UNIX or Linux systems

Eng neer w assemble the backend accounting system for an online game compan, This system w nc ude database entries for a customers for both one time and recurring pay ments System must a so ca cu ate royalty payments based on data gathered from the backend game server

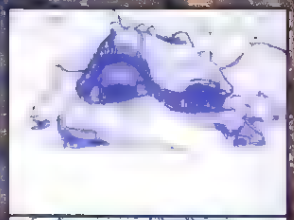
Eng neer w a so create the software for a web based community s te nc ud ng personal porta s personal web pages ema and calendars for a customers This web based community s te must be ab e to gather data from the game server

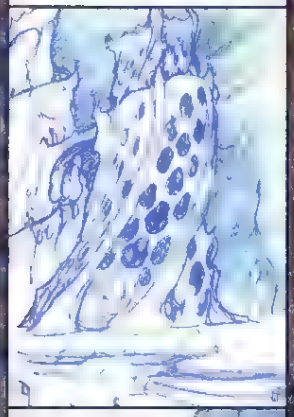
Eng neer must work we w th a team of other program mers Fam ar ty w th MUDs and LPC or Pike a p us

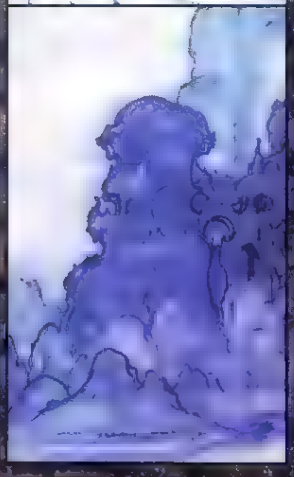
For all positions Applicant must be ready to relocate to the San Francisco Bay Area

Contact Info. Qua fied candidates shou d ema re sumes and cover etter n Word or ASC text format to job s kot net or fax to 5 0 649 4034 Pr nc pa s only Not a contract post on US or Canad an c t zens or res dents on y No phone calls please Equal opportun ty employer

High Fantasy.
High Tech.
High Art.
Hiring.







What you see is the unfolding of a dream. Give life to characters who think, feel, and speak. Join us as we make our high concept a reality.

Now Interviewing:

- Art Director
- Storyboard Artist
- Character Designer
- Photoshop/Illustrator Artist
- Modeling Technical Director
- Lighting Technical Director
- Personality/Behavior Scripter
- Lead Character Animator-Maya 3.5
- Associate Character Animators
- Effects Animator
- Animators
- Production Manager

During the Game Developers Conference
Leave your résumé and VHS reel or portfolio at the front desk of the San José Convention Inn
455 South Second Street
San José, California 95113
attention Jordan Wong

After the Conference
Send your résumé, VHS reel or portfolio, and a cover letter to Artware, 99 Main Street
Nyack, New York 10960
attention Jordan Wong

Artware.
Games for the Rest of Us.

VETERAN GAMERS PS2 TITLES

JOB FAIR BOOTH #1834

Our Cool Publisher Clients
EA Sports
Mattel
Lego
AOL

Our Cool Games
NASCAR 2000
Pool of Radiance II
Madden Football
Star Trek: DS9
On-line Titles

Our Cool Place
Fun environment with awesome talent. Located in scenic Marin, just north of San Francisco

"Are You Cool Enough?"
Senior or Lead Console Programmers
Senior or Lead PC Programmers
Producers or Associate Producers
Art Directors and Lead Artists
2D/3D Artists & 3D Animators
Entry Programmers
Interactive Writers

Resume: Stormfront Studios, Attn: M. Daglow
4040 Civic Center Dr., San Rafael, CA 94903
Fax: (415) 461-3865
email: mdaglow@earthlink.net
www.stormfrontstudios.com

STORMFRONT STUDIOS

CALL FOR WRITERS

Dear Professional Game Developer

If you're technically astute and have a way with words *Game Developer* magazine needs you. We're looking for feature articles on programming (graphics, AI, networking, and so on), art and animation techniques, game design, audio technology, testing and QA issues, and other relevant technical topics. We want to cover topics for variety of platforms, including the PC, console, arcade, and handhelds. Give back to the community some of the hard-won knowledge that you've earned!

We're also on the prowl for postmortems of recently completed games, always looking for people who are interested in reviewing game development tools, and constantly hunting down rabid "Soapbox" column writers.

Please send your article abstract, outline, or wacky idea to [redacted].

Writer's guidelines can be downloaded from our web site at www.gdmag.com/writguid.htm

Thanks

Alex Dunne, Editorial Director

CMP Game DEVELOPER

Want to get into the game? Get into NVIDIA.

If you're looking for a way to take your "gaming" experience to the next level, there's a good chance we've got the positions you need to see now. As the world's leading supplier of performance 3D graphics processors, we're working on stuff that has to be seen to be believed. We offer very competitive salaries and benefits—and we've been know to have some serious fun too. This is a great opportunity to join the fastest-growing 3D company. Apply online at www.nvidia.com. Or e-mail your resume to: hr@nvidia.com. You can also fax it to: 408-615-2800. If you want to get more out of your experience, you need to get into NVIDIA.

- *SALES*
- *MARKETING*
- *HARDWARE ENGINEERING*
- *SOFTWARE ENGINEERING*
- *SYSTEM ENGINEERING*
- *ARCHITECTURE*
- *QA*

nVIDIA™

graphics

© 2000 NVIDIA Corporation. All rights reserved. NVIDIA and the NVIDIA logo are trademarks of NVIDIA Corporation.

• United States • Canada • Mexico • France • Netherlands • United Kingdom • Spain • Canada • Mexico • Netherlands • United Kingdom • Spain •

A PARTNER FOR ALL YOUR MEDIA NEEDS

cinram

Service you can depend on since 1969

CD-AUDIO
CD-ROM
CD-VIDEO
DVD-VIDEO
DVD-ROM
VHS CASSETTES
AUDIO CASSETTES

PREMASTERING
MASTERING
REPLICATION
SCREEN PRINTING
OFFSET PRINTING
DVD AUTHORING
PACKAGING DESIGN
FULFILLMENT

www.cinram.com
1-800-433-3472

• Spain • United Kingdom • Canada • Mexico • France • United States • Netherlands • United Kingdom • Spain •

SCENE GENIE™

Sophisticated Animation, 3D Matching and Motion Capture

for

3D Studio MAX

afx.com

vancouver film school

VFS 3D Animation Instructor Nick Michales
3D Digital Effects Designer Mercury Filmworks

40 WEEKS
27 ALL-NIGHTERS
3 HAIR COLORS
6 CHARACTER ANIMATIONS
7 (VIRTUAL) EXPLOSIONS
and
ONE KILLER DEMO REEL

3D Animation & Digital Effects
New Media • Classical Animation
Film • Acting for Film & Television
Writing for Film & Television
Make-up for Film & Television

VFS education
1-800-661-4101
400 West Hastings Street, Vancouver, BC Canada V6B 1L2
E-mail: q129@vfs.com

1000 Bulk CD's only .58¢ each

Also Providing services for:

- Shaped CD's
- Business Card CD's
- Printing
- Packaging
- Fulfillment
- Graphic Design
- 4x & 8x Blank CDR's

LIGHTNING HILL Replicators

Your partners in success!

Tel: (310) 827-7764 Fax: (310) 827-3744
or call toll free at 1 888 965-4448
email: sales@lightninghill.com

COMPUTER ARTS
VIDEO/FILM

BACHELOR of FINE ARTS
MASTER of ARTS
MASTER of FINE ARTS

Game Development
Interactive Design • Motion Graphics
Sound Design • Video/Film
2D Animation • 3D Animation

Silicon Graphics, Windows NT,
DEC Alpha and Power Macintosh

After Effects, Alias, Animo, Director,
Digital Studio, Houdini, Illustrator,
Flint, Lightwave, Maya, Motivate,
Nichiman Painter Photoshop,
PlayStation Net, Yaroze, Renderman,
Softimage, 3D Studio Max, and more

Savannah College
Art and Design
An International University for the Arts

P.O. Box 3146 • Savannah, GA 31402-3146 USA
1.800.869.SCAD
info@scad.edu • www.scad.edu • www.ca.scad.edu

Savannah College of Art and Design admits students of any race, color, and national or ethnic origin.

Image: Hyun Hun, Seoul, South Korea; Alias/Wavefront Maya

TREMOR
ENTERTAINMENT

Wanna shake things up?

- Programmers
- Artists
- Animators
- Designers

Burbank, California
(818) 729 0020
www.tremor.net

Forgive our enthusiasm, but we're
PSYCHED.

Available Positions

_Design

3D Modelers
3D Animators
Texture Artists
Web Developers
Content Engineers (TCL/Tk exp)

_Engineering

3D Workspace/Human Factors
Communications Development
3D Software Development
Software Development
Web Browser Development
Technical Writers

Do you think that the Internet could do with a facelift? Do you suffer from Windows rage?
Do you want your internet experience to feel as good as the games you play?

We do. Here at Muse Communications we are creating a platform that enables the creation and delivery of applications and Internet content that will make YOU excited. You're gonna dig this.... big time.

_muse communications corporation

1950 elkhorn court, san mateo, ca 94403
fax: (650) 378 - 6339 jobs@musecorp.com www.musecorp.com

BEYOND
TECHNOLOGY
BEYOND
EDUCATION
BEYOND
ENTERTAINMENT

BEYOND IMAGINATION

www.e3expo.com or 800-315-1133

E3 is a trade event. No one under 18 will be admitted (including infants). This policy is strictly enforced.

CONTINUED FROM PAGE 80

Hollywood is often held up as a standard by which we measure the progress of our own branch of the entertainment industry. The widely touted factoid that 1998's computer and videogame revenues surpassed Hollywood box office receipts is one example. When you consider the total revenues from video rentals, licensing fees, and advertising, Hollywood is still way ahead — but our growth rate remains much higher than Hollywood's, and shows little sign of slowing. It's possible to make a good case that at some point in the future, linear media such as film and television will be considered a subset of interactive entertainment, just like how at first there were just movies, then "talkies" were introduced, and eventually "talkies" became "movies" and the old movies became "silent movies." Perhaps what we now call movies will one day be called "linears" or "branchless movies." If that sounds ambitious now, at the 1988 CGDC it would have been wild fantasy. Now, keeping our rate of growth in mind, even more seems possible.

And so if it takes another ten years for Sid to get a chance to sleep in the Lincoln Bedroom (and scrawl "Lee Rules!" on the wall?) just remember you heard it here first. After that, who knows...the Nobel prize for interactive literature? ■

ADVERTISER INDEX

NAME	PAGE	NAME	PAGE
AliasIWavefront	C3	Midway Games	69
Artware	73	Motek	11
Ascension Technology Corp.	14	Multigen	51
Aureal Semiconductor	17	Muse Corp.	77
Autonomous Effects	76	n-Space	69
BOPS Inc.	21	Newtek Inc.	61
Bungie	71	Nintendo of America	53, 60
Cibro Technologies	64	Numerical Design Ltd.	6
Cinram	76	Nvidia	75
Conitec Datasystems Inc.	79	NxN Digital Entertainment	C2
The Coriolis Group	69	Proksim Software	13
Dice.com	71	RAD Game Tools	C4
Digimation	59	Retro Studios	29
Discreet	33	S3 Diamond	63
E-Lysium Transaction Systems	45	Savannah College of Art & Design	77
Giant Studios	25	Sensaura CRL	3
Global Majic Software	49	Skotos Tech Inc.	72
Hewlett Packard	39	SN Systems	9
High Voltage Software	71	Sony DADC	35
Ikarion Software GmbH	71	Sony Online	70
Inoiz.com	37	ST Microelectronics	67
Interact Source	68	Staccato	22
LicenseMusic.com	57	Stormfront Studios	74
Lightning Hill Replicators	76	Telekinesys	5
LIPSinc	41	Tremor Entertainment	77
Macrovision	27	Vancouver Film School	76
MathEngine	47	Viewpoint Digital	19

Create cool 3-D Applications!
GameStudio
Authoring System for 3D Realtime Applications
◆ Interactive 3D applications without programming!
◆ Create games, flight simulators, virtual exhibitions...
◆ Contains World + Model Editor and the A4 engine
◆ 3D action, adventure, role playing games included
◆ 6 degrees of freedom, mip and shadow mapping
◆ Particles, flare effects, fog, static + dynamic light sources
◆ D3D and software mode, 8, 16, 32 bit screen resolution
◆ Extremely fast rendering, even without 3D hardware
◆ 2D engine for menus, overlays, buttons, sprites, text
◆ Multiplayer via split screen, modem, network, Internet
◆ Cool scripting language for AI and user interface
◆ Imports 3DS, MAP, MDL, PCX, BMP, WAV, MID, AVI
New Version 4.1
3D GameStudio Commercial ... $199
3D GameStudio Professional ... $1250
(+Multiplayer+ CD-Audio + AVI-Player +Database Input)
Infos, demos, ordering → http://www.conitec.com
CONITEC
1951 4th Ave, Ste 301 ■ San Diego CA 92101
Tel (619) 702-4420 ■ Fax 702-4419 ■ www.conitec.com

by Noah Falstein

A Tale of Two GDCs

NEWS FLASH: Sid Meier will be one of the Kennedy Center Honorees next year and is invited to spend an evening at the White House!

O.K., I'm making it up. But it didn't sound entirely far-fetched, did it? After all, if movie directors and rock stars can be honored, why not Sid? He's certainly deserving when measured by hours of entertainment provided. More than deserving if quality counts, too.

But our industry is still too young to achieve that level of recognition — for now. If your reaction was complete disbelief, just give it some time.

Recently I offered to write this article about the interactive entertainment industry "coming of age." But it's too easy to make the case for putting an end to all those articles that begin "Although the computer game industry is still in its infancy..." At the very least we're well into adolescence, as the success of Lara Croft attests.

Then I had the good fortune to speak at the first Australian Game Developers Conference in Sydney. I was coming out for a week of work with Microforte, a game developer that was also co-sponsoring the conference, so combining the two seemed logical.

I was also one of the 180 or so attendees at the second U.S. CGDC (now known simply as the GDC) in 1988 — not one of the two dozen people Chris Crawford invited to meet in his living room, but the first conference at a hotel. The similarities and differences between the Australian GDC and my first CGDC convinced me just how far we've come, as well as how far we have to go.

There were around 250 attendees at this first AGDC, more than they expected, and all the more significant considering Australia's population. A proportional attendance in the U.S. would be closer to 3,500 — the size of the CGDC just a few years ago. Back in 1988, it was impressive simply to have 180 game developers all in one place. The Australian conference was much more professionally organized, too, borrowing a great deal from what has been refined through years of experience in the U.S.

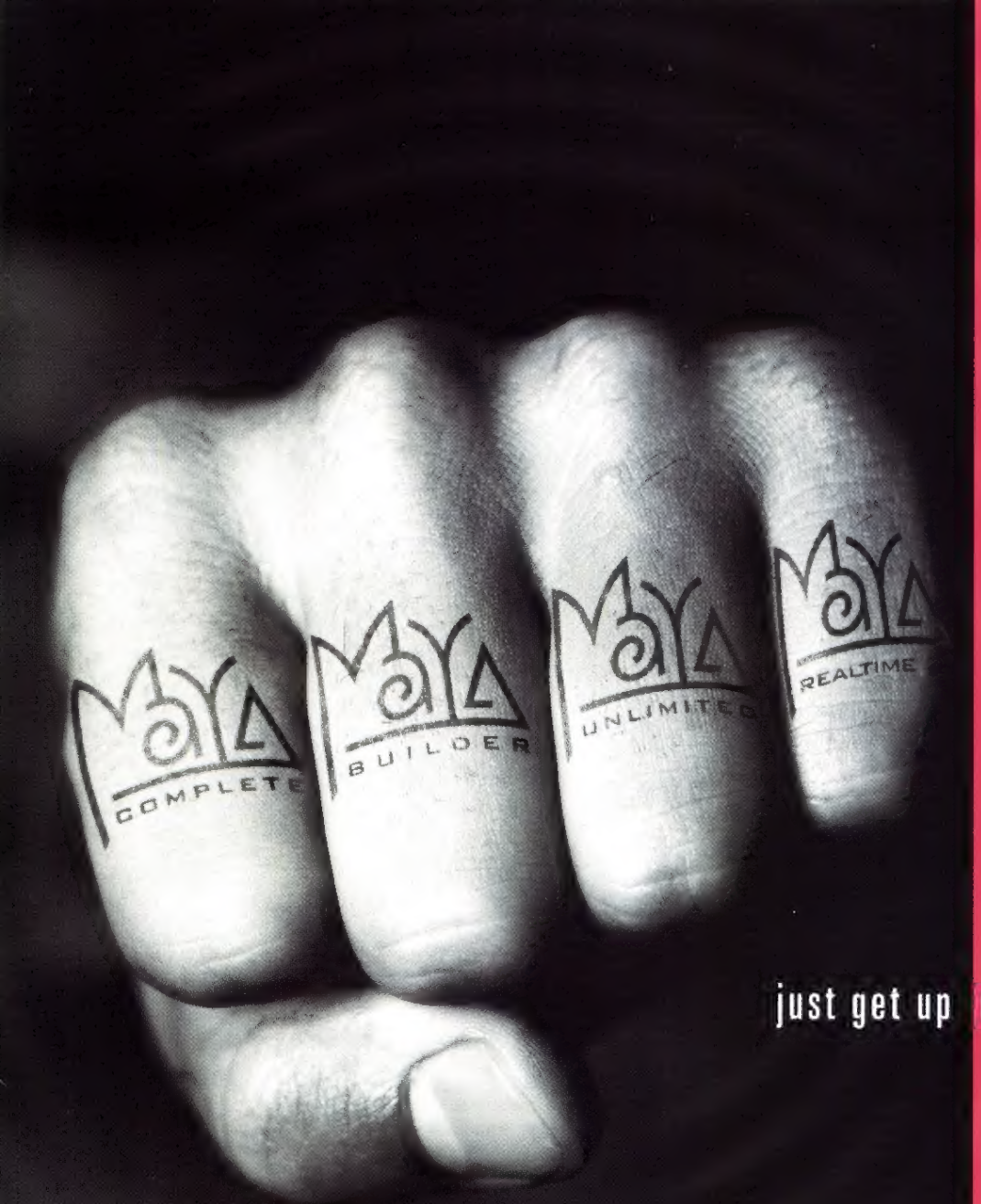
illustration by Jackie Urbanovic

But more striking than the differences were some of the similarities between the AGDC and that early CGDC. The large majority of the attendees at both were under 30 — not unusual in this industry, but it seemed the average age was a good five to seven years younger than the 1999 GDC. There was an air of enthusiasm and optimism once common in the U.S. and now increasingly rare, no doubt due to the increasingly cynical and skeptical "old-timers" like me. There were several times when I cautioned impassioned young developers against some proposed course of game development with some variation on "that's been tried repeatedly before and failed in these ways..." I think I saved some people considerable heartbreak, but there's poignancy in realizing that we've explored some frontiers and found nothing but badlands. It's more fun when every horizon is virgin territory, full of promise, and sometimes the excitement of being an explorer is worth the attendant pain.

Another measure of how far the industry has come was the very fact that I was at the conference in my capacity as freelance game designer. The idea of flying someone out for a week of work from more than 7,000 miles away would have been fanciful 12 years ago and is rare even now, but to my delight is becoming more common. It seems the increasingly large and robust interactive industry has grown to the point of being able to support a growing group of freelancers similar to Hollywood's ranks of writers, actors, and technicians.

CONTINUED ON PAGE 79

Noah Falstein runs The Inspiracy, providing freelance game design, producing, and writing for both established and up-and-coming companies. When he isn't jet-setting around the world he can be found in Marin County, Calif. For contact info see his web site at http://www.theinspiracy.com.



sometimes,

when you least expect it,

an idea can hit you

like a suckerpunch.

you never even see it coming.

you've been hit with ideas before,

but never one so powerful.

it may have even knocked you on your ass.

no worries though.

just get up dust yourself off

and know that

you have the tools

to wrestle it to the ground.

Maya for only $3,995. Imagine that!

For a limited time 3D Studio Max users can purchase their first seat of Maya Complete for only $3,995 USD.
Visit **www.aliaswavefront.com/entertainment** or call 1.800.447.2542.

Alias | wavefront

Offer valid for all Max users with current version. Offer valid until May 31, 2000 on your first seat of Maya Complete. Prices shown in US dollars and may vary internationally.

Maya™ CAN YOU IMAGINE

The Best in Game Development Technology!

Granny Character Animation

Introducing Granny - the most sophisticated character animation system available. **Smoothly animate** your characters blending from a run to a walk to a sliding stop, all while using **real-time inverse kinematics** to point or look at other objects. **Deform clothing** to fit your different characters. Perfect exporting with built-in Bink **texture compression**! Much more!

Bink Video Technology

Bink - the only true-color codec good enough for games! Incredible quality (**better than DVD**/MPEG II). Up to **three times faster** than other true-color codecs. Perceptibly lossless, **8 to 1 audio compression**. Decompresses to DirectDraw, DIBSections, YUV overlays, or any other piece of memory. Available for Win32 (MacOS coming). Think Bink!

Smacker Video Technology

Smacker - the best 256-color codec on the planet! Smacker is the best codec for: **256-color games** (of course), games targeting Pentium 133s and below, games needing video sprites or video transparency (which is **much faster in 256 colors**), or games with very high-resolution videos (800x600 or 1024x768). Smacker is available on Win32, Win16, DOS, and MacOS.

Miles Sound System

Miles - the ultimate sound system! Miles supports 2D audio, 3D audio, MIDI with DLS, streaming, cd audio (red book), **DSP filtering**, **MP3 playback (including patent rights)**, **Internet voice chat** (2900, 2400, and 1200 bits per second codecs), Creative **EAX 1 and 2**, Aureal **A3D 1 and 2**, RSX 3D, DirectX 7, Dolby Surround, QSound, **software EAX emulation**, and more!

Visit us at the Game Developer Conference, Booth #826

NO ROYALTIES WHATSOEVER! OVER 3,000 GAMES USE OUR TECHNOLOGIES!

Call today or visit our web site for more info:

425-893-4300 www.radgametools.com

RAD GAME TOOLS ©

Made with love by

Our goal is to preserve classic video game magazines so that they are not lost permanently.

People interested in helping out in any capacity, please visit us at retromags.com.

No profit is made from these scans, nor do we offer anything available from the publishers themselves.

If you come across anyone selling releases from this site, please do not support them and do let us know.

Thank you!